

5

APPENDIX

Program Listing Pages 1-162

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162

Option Explicit

```
Private mstrFullName As String
Private mstrAbrvName As String
Private mstrFullNameTrim As String
Private mstrAbrvNameTrim As String
```

```
Property Get FullName() As String
    FullName = mstrFullName
End Property
```

```
Property Let FullName(ByVal Source As String)
    mstrFullName = Source
    mstrFullNameTrim = SuperTrim(Source)
End Property
```

```
Property Get FullNameTrim() As String
    FullNameTrim = mstrFullNameTrim
End Property
```

```
Property Get AbrvName() As String
    AbrvName = mstrAbrvName
End Property
```

```
Property Let AbrvName(ByVal Source As String)
    mstrAbrvName = Source
    mstrAbrvNameTrim = SuperTrim(Source)
End Property
```

```
Property Get AbrvNameTrim() As String
    AbrvNameTrim = mstrAbrvNameTrim
End Property
```

```
Public Function HasAbbreviation() As Boolean
    If mstrFullName <> mstrAbrvName Then HasAbbreviation = True
End Function
```

```

    mstrFullName = Source
    mstrAbrvName = Source
    mstrFullNameTrim = SuperTrim(Source)
    mstrAbrvNameTrim = SuperTrim(Source)
    HasAbbreviation = (mstrFullName <> mstrAbrvName)
End Function

```

'-- Copyright 2001 by Robert L. Jackson.

Option Explicit

Private mcolAbbreviations As Collection

```
Public Enum bcAbbreviationType
    Table6
    Table10
    All
End Enum
```

```
Private Sub Class_Initialize()
    Set mcolAbbreviations = New Collection
End Sub
```

```
Public Function Count() As Long
    Count = mcolAbbreviations.Count
End Function
```

```
Public Function Item(Index As Variant) As cAbbreviation
    Set Item = mcolAbbreviations.Item(Index)
End Function
```

```
Public Function NewEnum() As IUnknown
    Set NewEnum = mcolAbbreviations.[_NewEnum]
End Function
```

```
Public Sub LoadData(Source As bcAbbreviationType)
```

```
    If Source = Table6 Or Source = All Then
```

```
        Add "Administrative", "Admin."
        Add "Administration", "Admin."
        Add "Administrations", "Admins."
        Add "Administrator", "Adm'r"
        Add "Administrators", "Adm'rs"
        Add "Administratrix", "Adm'x"
        Add "Administratrixes", "Adm'xs"
        Add "Advertising", "Adver."
        Add "Agriculture", "Agric."
        Add "Agricultural", "Argic."
        Add "America", "Am."
        Add "Americas", "Ams."
        Add "American", "Am."
        Add "Americans", "Ams."
        Add "Associate", "Assoc."
        Add "Associates", "Assocs."
        Add "Association", "Ass'n"
        Add "Associations", "Ass'ns"
        Add "Atlantic", "Atl."
        Add "Authority", "Auth."
        Add "Authorities", "Auths."
        Add "Automobile", "Auto."
        Add "Automobiles", "Autos."
        Add "Avenue", "Ave."
        Add "Avenues", "Aves."
        Add "Bankruptcy", "Bankr."
        Add "Bankruptcies", "Bankrs."
        Add "Board", "Bd."
        Add "Broadcast", "Broad."
        Add "Broadcasts", "Broads."
        Add "Broadcasting", "Broad."
        Add "Brotherhood", "Bhd."
        Add "Brotherhoods", "Bhds."
        Add "Brothers", "Bros."
        Add "Building", "Bldg."
        Add "Buildings", "Bldgs."
        Add "Business", "Bus."
        Add "Casualty", "Cas."
        Add "Center", "Ctr."
        Add "Centers", "Ctrs."
        Add "Centre", "Ctr."
        Add "Centres", "Ctrs."
        Add "Central", "Cent."
        Add "Chemical", "Chem."
        Add "Chemicals", "Chems."
        Add "Commission", "Comm'n"
        Add "Commissioner", "Comm'r"
        Add "Committee", "Comm."
        Add "Committees", "Comms."
        Add "Company", "Co."
        Add "Companies", "Cos."
        Add "Consolidated", "Consol."
        Add "Construction", "Constr."
        Add "Cooperative", "Coop."
        Add "Cooperatives", "Coops."
        Add "Corporation", "Corp."
        Add "Corporation", "Corps."
        Add "Department", "Dep't"
        Add "Departments", "Dep'ts"
        Add "Developer", "Dev."
        Add "Development", "Dev."
        Add "Developers", "Devs."
        Add "Director", "Dir."
        Add "Directors", "Dir."
        Add "Distributor", "Distrib."
        Add "Distributors", "Distribs."
        Add "Distributing", "Distrib."
        Add "District", "Dist."
        Add "Districts", "Dists."
        Add "Division", "Div."
```

Add "Divisions", "Divs."
 Add "East", "E."
 Add "Eastern", "E."
 Add "Economic", "Econ."
 Add "Economical", "Econ."
 Add "Economy", "Econ."
 Add "Economies", "Econs."
 Add "Education", "Educ."
 Add "Educational", "Educ."
 Add "Electric", "Elec."
 Add "Electrical", "Elec."
 Add "Electricity", "Elec."
 Add "Electronic", "Elec."
 Add "Electronics", "Elects."
 Add "Engineer", "Eng'r"
 Add "Engineers", "Eng'rs"
 Add "Engineering", "Eng'g"
 Add "Enterprise", "Enter."
 Add "Enterprises", "Enter."
 Add "Environment", "Env't"
 Add "Environments", "Env'ts"
 Add "Environmental", "Envtl."
 Add "Equality", "Equal."
 Add "Equipment", "Equip."
 Add "Equipments", "Equips."
 Add "Examiner", "Exam'r"
 Add "Examiners", "Exam'rs"
 Add "Exchange", "Exch."
 Add "Exchanges", "Exchs."
 Add "Executor", "Ex'r"
 Add "Executors", "Ex'rs"
 Add "Executrix", "Ex'x"
 Add "Executrices", "Ex'xs"
 Add "Federal", "Fed."
 Add "Federation", "Fed'n"
 Add "Federations", "Fed'ns"
 Add "Finance", "Fin."
 Add "Finances", "Fins."
 Add "Financial", "Fin."
 Add "Financing", "Fin."
 Add "Foundation", "Found."
 Add "Foundations", "Founds."
 Add "General", "Gen."
 Add "Generals", "Gens."
 Add "Guaranty", "Guar."
 Add "Guaranties", "Guars."
 Add "Hospital", "Hosp."
 Add "Hospitals", "Hosps."
 Add "Housing", "Hous."
 Add "Incorporated", "Inc."
 Add "Indemnity", "Indem."
 Add "Indemnities", "Indems."
 Add "Independent", "Indep."
 Add "Industry", "Indus."
 Add "Industries", "Indus."
 Add "Industrial", "Indus."
 Add "Information", "Info."
 Add "Institute", "Inst."
 Add "Institutes", "Insts."
 Add "Institution", "Inst."
 Add "Institutions", "Insts."
 Add "Insurance", "Ins."
 Add "International", "Int'l"
 Add "Internationals", "Int'ls"
 Add "Investment", "Inv."
 Add "Investments", "Inv."
 Add "Laboratory", "Lab."
 Add "Laboratories", "Lab."
 Add "Liability", "Liab."
 Add "Liabilities", "Liabs."
 Add "Limited", "Ltd."
 Add "Litigation", "Litig."
 Add "Machine", "Mach."
 Add "Machines", "Machs."
 Add "Machinery", "Mach."
 Add "Manufacturer", "Mfr."
 Add "Manufacturers", "Mfrs."
 Add "Manufacturing", "Mfg."
 Add "Market", "Mkt."
 Add "Markets", "Mkts."
 Add "Marketing", "Mktg."
 Add "Medical", "Med."
 Add "Medicine", "Med."
 Add "Medicines", "Meds."
 Add "Memorial", "Mem'l"
 Add "Memorials", "Mem'ls"
 Add "Metropolitan", "Metro."
 Add "Municipal", "Mun."
 Add "Mutual", "Mut."
 Add "National", "Nat'l"
 Add "North", "N."
 Add "Northern", "N."
 Add "Organization", "Org."
 Add "Organizations", "Orgs."
 Add "Organizing", "Org."
 Add "Pacific", "Pac."
 Add "Pharmaceuticals", "Pharm."
 Add "Pharmaceutical", "Pharm."
 Add "Product", "Prod."
 Add "Products", "Prods."
 Add "Production", "Prod."

```

Add "Productions", "Prods."
Add "Professional", "P"
Add "Professionals", "P"
Add "Public", "Pub."
Add "Publishing", "Publ'g"
Add "Railroad", "R.R."
Add "Railroads", "R.Rs."
Add "Railway", "Ry."
Add "Railways", "Rys."
Add "Refining", "Ref."
Add "Regional", "Reg'l"
Add "Reproduction", "Reprod."
Add "Reproductions", "Reprods."
Add "Reproductive", "Reprod."
Add "Road", "Rd."
Add "Roads", "Rds."
Add "Savings", "Sav."
Add "School", "Sch."
Add "Schools", "Sch."
Add "Security", "Sec."
Add "Securities", "Sec."
Add "Service", "Serv."
Add "Services", "Servs."
Add "Social", "Soc."
Add "Socials", "Socs."
Add "Society", "Soc'y"
Add "Societies", "Soc'ys"
Add "South", "S."
Add "Southern", "S."
Add "Steamship", "S.S."
Add "Steamships", "S.S."
Add "Street", "St."
Add "Streets", "Sts."
Add "Subcommittee", "Subcomm."
Add "Subcommittees", "Subcomms."
Add "Surety", "Sur."
Add "Sureties", "Surs."
Add "System", "Sys."
Add "Systems", "Sys."
Add "Technology", "Tech."
Add "Technologies", "Techs."
Add "Telecommunication", "Telecomm."
Add "Telecommunications", "Telecomms."
Add "Telephone", "Tel."
Add "Telephones", "Tels."
Add "Telegraph", "Tel."
Add "Telegraphs", "Tels."
Add "Temporary", "Temp"
Add "Transcontinental", "Transcon."
Add "Transport", "Transp."
Add "Transports", "Transps."
Add "Transportation", "Transp."
Add "Uniform", "Unif."
Add "Uniforms", "Unifs."
Add "University", "Univ."
Add "Universities", "Univs."
Add "Utility", "Util."
Add "Utilities", "Utils."
Add "West", "W."
Add "Western", "W."

```

```

End If

```

```

If Source = Table10 Or Source = All Then

```

```

Add "Alabama", "Ala."
Add "Alaska", "Alaska"
Add "American Samoa", "Am. Sam."
Add "Arizona", "Ariz."
Add "Arkansas", "Ark."
Add "California", "Cal."
Add "Canal Zone", "C.Z."
Add "Colorado", "Colo."
Add "Connecticut", "Conn."
Add "Delaware", "Del."
Add "District of Columbia", "D.C."
Add "Florida", "Fla."
Add "Georgia", "Ga."
Add "Guam", "Guam"
Add "Hawaii", "Haw."
Add "Idaho", "Idaho"
Add "Illinois", "Ill."
Add "Indiana", "Ind."
Add "Iowa", "Iowa"
Add "Kansas", "Kan."
Add "Kentucky", "Ky."
Add "Los Angeles", "L.A."
Add "Louisiana", "La."
Add "Maine", "Me."
Add "Maryland", "Md."
Add "Massachusetts", "Mass."
Add "Michigan", "Mich."
Add "Minnesota", "Minn."
Add "Mississippi", "Miss."
Add "Missouri", "Mo."
Add "Montana", "Mont."
Add "Nebraska", "Neb."
Add "Nevada", "Nev."
Add "New Hampshire", "N.H."
Add "New Jersey", "N.J."
Add "New Mexico", "N.M."

```

Add "New York", "N.Y."
 Add "North Carolina", "N.C."
 Add "North Dakota", "N.D."
 Add "Northern Mariana Islands", "N. Mar. I."
 Add "Ohio", "Ohio"
 Add "Oklahoma", "Okla."
 Add "Oregon", "Or."
 Add "Pennsylvania", "Pa."
 Add "Philadelphia", "Phila."
 Add "Puerto Rico", "P.R."
 Add "Rhode Island", "R.I."
 Add "South Carolina", "S.C."
 Add "South Dakota", "S.D."
 Add "Tennessee", "Tenn."
 Add "Texas", "Tex."
 Add "Utah", "Utah"
 Add "Vermont", "Vt."
 Add "Virgin Islands", "V.I."
 Add "Virginia", "Va."
 Add "Washington", "Wash."
 Add "West Virginia", "W. Va."
 Add "Wisconsin", "Wis."
 Add "Wyoming", "Wyo."
 Add "New South Wales", "N.S.W."
 Add "Queensland", "Queensl."
 Add "South Australia", "S. Austrl."
 Add "Tasmania", "Tas."
 Add "Victoria", "Vict."
 Add "West Australia", "W. Austrl."
 Add "Alberta", "Alta."
 Add "British Columbia", "B.S."
 Add "Manitoba", "Man."
 Add "New Brunswick", "N.B."
 Add "Newfoundland", "Nfld."
 Add "Northwest Territories", "N.W.T."
 Add "Nova Scotia", "N.S."
 Add "Ontario", "Ont."
 Add "Prince Edward Island", "P.E.I."
 Add "Québec", "Que."
 Add "Saskatchewan", "Sask."
 Add "Yukon", "Yuokn"
 Add "Afghanistan", "Afg."
 Add "Africa", "Afr."
 Add "Albania", "Alb."
 Add "Algeria", "Alg."
 Add "Andorra", "Andorra"
 Add "Angola", "Angl."
 Add "Anguilla", "Anguilla"
 Add "Antigua & Barbuda", "Ant. & Barb."
 Add "Argentina", "Arg."
 Add "Armenia", "Arm."
 Add "Australia", "Austl."
 Add "Austria", "Aus."
 Add "Azerbaijan", "Azer."
 Add "Bahamas", "Bah."
 Add "Bahrain", "Bahr."
 Add "Bangladesh", "Bangl."
 Add "Barbados", "Barb."
 Add "Belarus", "Belr."
 Add "Belgium", "Belg."
 Add "Belize", "Belize"
 Add "Benin", "Benin"
 Add "Bermuda", "Berm."
 Add "Bhutan", "Bhutan"
 Add "Bolivia", "Bol."
 Add "Bosnia & Herzegovina", "Bosn. & Herz."
 Add "Botswana", "Bots."
 Add "Brazil", "Braz."
 Add "Brunei", "Brunei"
 Add "Bulgaria", "Bulg."
 Add "Burkina Faso", "Burk. Faso"
 Add "Burundi", "Burundi"
 Add "Cambodia", "Cambodia"
 Add "Cameroon", "Cameroon"
 Add "Canada", "Can."
 Add "Cape Verde", "Cape Verde"
 Add "Cayman Islands", "Cayman Is."
 Add "Central African Republic", "Cent. Afr. Rep."
 Add "Chad", "Chad"
 Add "Chile", "Chile"
 Add "People's Republic of China", "P.R.C."
 Add "Colombia", "Colom."
 Add "Comoros", "Comoros"
 Add "Congo", "Congo"
 Add "Costa Rica", "Costa Rica"
 Add "Côte d'Ivoire", "Côte d'Ivoire"
 Add "Croatia", "Croat."
 Add "Cuba", "Cuba"
 Add "Cyprus", "Cyprus"
 Add "Czech Republic", "Czech Rep."
 Add "Denmark", "Den."
 Add "Djibouti", "Djib."
 Add "Dominica", "Dominica"
 Add "Dominican Republic", "Dom. Rep."
 Add "Ecuador", "Ecuador"
 Add "Egypt", "Egypt"
 Add "El Salvador", "El Sal."
 Add "England", "Eng."
 Add "Equatorial Guinea", "Eq. Guinea"
 Add "Eritrea", "Eri."
 Add "Estonia", "Est."

Add "Ethiopia", "Eth."
Add "Europe", "Eur."
Add "Falkland Islands", "Falkland Is."
Add "Fiji", "Fiji"
Add "Finland", "Fin."
Add "France", "Fr."
Add "Cabon", "Gabon"
Add "Gambia", "Gam."
Add "Georgia", "Geor."
Add "Germany", "F.R.G."
Add "Federal Republic of Germany", "F.R.G."
Add "Ghana", "Ghana"
Add "Gibraltar", "Gib."
Add "Great Britain", "Gr. Brit."
Add "Greece", "Greece"
Add "Grenada", "Gren."
Add "Guatemala", "Guat."
Add "Guinea", "Guinea"
Add "Guinea-Bissau", "Guinea-Bissau"
Add "Guyana", "Guy."
Add "Haiti", "Haiti"
Add "Honduras", "Hond."
Add "Hong Kong", "H.K."
Add "Hungary", "Hung."
Add "Iceland", "Ice."
Add "India", "India"
Add "Indonesia", "Indon."
Add "Iran", "Iran"
Add "Iraq", "Iraq"
Add "Ireland", "Ir."
Add "Israel", "Isr."
Add "Italy", "Italy"
Add "Jamaica", "Jam."
Add "Japan", "Japan"
Add "Jordan", "Jordan"
Add "Kazakhstan", "Kaz."
Add "Kenya", "Kenya"
Add "Kiribati", "Kiribati"
Add "Kuwait", "Kuwait"
Add "Kyrgyzstan", "Kyrg."
Add "Laos", "Laos"
Add "Latvia", "Lat."
Add "Lebanon", "Leb."
Add "Lesotho", "Lesotho"
Add "Liberia", "Liber."
Add "Libya", "Libya"
Add "Liechtenstein", "Liech."
Add "Lithuania", "Lith."
Add "Macedonia", "Maced."
Add "Madagascar", "Madag."
Add "Malawi", "Malawi"
Add "Malaysia", "Malay."
Add "Maldives", "Maldives"
Add "Mali", "Mali"
Add "Malta", "Malta"
Add "Marshall Islands", "Marsh. Is."
Add "Mauritania", "Mauritania"
Add "Mauritius", "Mauritius"
Add "Mexico", "Mex."
Add "Micronesia", "Micr."
Add "Moldova", "Mold."
Add "Monaco", "Monaco"
Add "Mongolia", "Mong."
Add "Montserrat", "Montserrat"
Add "Morocco", "Morocco"
Add "Mozambique", "Mozam."
Add "Myanmar", "Myan."
Add "Namibia", "Namib."
Add "Mauru", "Mauru"
Add "Nepal", "Nepal"
Add "Netherlands", "Neth."
Add "New Zealand", "N.Z."
Add "Nicaragua", "Nicar."
Add "Niger", "Niger"
Add "Nigeria", "Nig."
Add "North America", "N. Am."
Add "Northern Ireland", "N. Ir."
Add "North Korea", "N. Korea"
Add "Norway", "Nor."
Add "Oman", "Oman"
Add "Pakistan", "Pak."
Add "Panama", "Pan."
Add "Papua New Guinea", "Papua N.G."
Add "Paraguay", "Para."
Add "Peru", "Peru"
Add "Philippines", "Phil."
Add "Pitcairn Island", "Pitcairn Is."
Add "Poland", "Pol."
Add "Portugal", "Port."
Add "Qatar", "Qatar"
Add "Romania", "Rom."
Add "Russia", "Russ."
Add "Rwanda", "Rwanda"
Add "St. Kitts & Nevis", "St. Kitts & Nevis"
Add "St. Helena", "St. Helena"
Add "St. Lucia", "St. Lucia"
Add "St. Vincent & the Grenadines", "St. Vincent"
Add "San Marino", "San Marino"
Add "São Tomé and Príncipe", "São Tomé & Príncipe"
Add "Saudi Arabia", "Saudi Arabia"
Add "Scotland", "Scot."

```

Add "Senegal", "Sen."
Add "Seychelles", "Sey."
Add "Sierra Leone", "Sierra"
Add "Singapore", "Sing."
Add "Slovakia", "Slovk."
Add "Slovenia", "Slovn."
Add "Solomo Islands", "Solom. Is."
Add "Somalia", "Somal."
Add "South Africa", "S. Afr."
Add "South Korea", "S. Korea"
Add "Spain", "Spain"
Add "Sri Lanka", "Sri Lanka"
Add "Sudan", "Sudan"
Add "Suriname", "Surin."
Add "Swaziland", "Swaz."
Add "Sweden", "Swed."
Add "Switzerland", "Switz."
Add "Syria", "Syria"
Add "Taiwan", "Taiwan"
Add "Tajikistan", "Taj."
Add "Thailand", "Thail."
Add "Togo", "Togo"
Add "Tonga", "Tonga"
Add "Trinidad and Tobago", "Trin. & Tobago"
Add "Tunisia", "Tunis."
Add "Turkey", "Turk."
Add "Turkmenistan", "Turkm."
Add "Turks and Caicos Islands", "Turks & Caicos Is."
Add "Tuvalu", "Tuvalu"
Add "Uganda", "Uganda"
Add "Ukraine", "Ukr."
Add "United Arab Emirates", "U.A.E."
Add "United Kingdom", "U.K."
Add "United States", "U.S."
Add "Uruguay", "Uru."
Add "Uzbekistan", "Uzb."
Add "Vanuatu", "Vanuatu"
Add "Vatican City", "Vatican"
Add "Venezuela", "Venez."
Add "Vietnam", "Vietnam"
Add "Brittish Virgin Islands", "Virgin Is."
Add "Wales", "Wales"
Add "Western Samoa", "W. Samoa"
Add "Yemen", "Yemen"
Add "Yugoslavia", "Yugo."
Add "Zaire", "Zaire"
Add "Zambia", "Zambia"
Add "Zimbabwe", "Zimb."

```

```

End If

```

```

End Sub

```

```

Friend Sub Add(FullName As String, AbrvName As String, Optional ByVal Key As Variant)

```

```

    Dim bcAbbrNew As cAbbreviation

```

```

    Set bcAbbrNew = New cAbbreviation

```

```

    With bcAbbrNew

```

```

        .FullName = FullName

```

```

        .AbrvName = AbrvName

```

```

    End With

```

```

    mcolAbbreviations.Add bcAbbrNew, Key

```

```

End Sub

```


cBlueCheckInstance - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Private mmswOriginalSelection As Word.Range

Public Sub Start()

```
Dim bcCitationBuilder As cCitationBuilder
Dim bcTimer As New cTimer
Dim lngStartFrom As Long
```

```
If fOptions.ForCourt = "" Then
    Call ShowOptions
    If fOptions.ForCourt = "" Then Exit Sub
End If
```

```
Set ErrorForm = New cErrorForm
Set mmswOriginalSelection = Word.Selection.Range
```

```
Set ActiveDoc = New cDocument
Call ActiveDoc.Initialize(Word.ActiveDocument)
```

```
lngStartFrom = 1
```

StartOver:

```
ErrorForm.Result = None
```

```
Set bcCitationBuilder = New cCitationBuilder
Set Citation = bcCitationBuilder.NextCitationFromWord(lngStartFrom)
```

Do While Not (Citation Is Nothing)

```
Call CheckCaseName
If ErrorForm.Result = Cancel Then GoTo Cancel Else If ErrorForm.Result = Change Then GoTo StartOver
```

```
Call CheckReporter
If ErrorForm.Result = Cancel Then GoTo Cancel Else If ErrorForm.Result = Change Then GoTo StartOver
```

```
Call CheckEditor
If ErrorForm.Result = Cancel Then GoTo Cancel Else If ErrorForm.Result = Change Then GoTo StartOver
```

```
Call CheckJurAndCourt
If ErrorForm.Result = Cancel Then GoTo Cancel Else If ErrorForm.Result = Change Then GoTo StartOver
```

```
Call CheckParallelReporters
If ErrorForm.Result = Cancel Then GoTo Cancel Else If ErrorForm.Result = Change Then GoTo StartOver
```

```
Call CheckDate
If ErrorForm.Result = Cancel Then GoTo Cancel Else If ErrorForm.Result = Change Then GoTo StartOver
```

```
Call CheckClauseOrder
If ErrorForm.Result = Cancel Then GoTo Cancel Else If ErrorForm.Result = Change Then GoTo StartOver
```

```
lngStartFrom = Citation.CitationEnd + 1
Set Citation = bcCitationBuilder.NextCitationFromWord(lngStartFrom)
```

Loop
Cancel:

```
Call EndBlueCheck
```

End Sub

Public Sub ShowOptions()

```
fOptions.Show vbModal
```

End Sub

Public Sub EndBlueCheck()

```
Dim strNumberOfChanges As String
```

```
ErrorForm.Hide
```

```
If Not ErrorForm.Result = Cancel Then
    Select Case ErrorForm.ChangesMade
```

```
        Case Is = 0
            strNumberOfChanges = "No changes were made."
```

```
        Case Is < 13
            strNumberOfChanges = Choose(ErrorForm.ChangesMade, "One change was made.", "Two changes were made.", "Three changes were made.", "Four changes were made.", "Five changes were made.", "Six changes were made.", "Seven changes were made.", "Eight changes were made.", "Nine changes were made.", "Ten changes were made.", "Eleven changes were made.", "Twelve changes were made.")
```

```
        Case Else
            strNumberOfChanges = ErrorForm.ChangesMade & " changes were made."
```

```
    End Select
```

```
    Call MsgBox("BlueCheck has finished checking this document. " & strNumberOfChanges, vbInformation, "BlueCheck")
```

```
End If
```

```
mmswOriginalSelection.Select
```

```
Clipboard.Clear
```

```
Set ErrorForm = Nothing
```

9

[illegible]

cErrorForm - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Private mlngChangesMade As Long

Private mbcBluebookRules As cMessages
Private mbcErrorMessages As cMessages

Private mstrErrorText As String
Private mstrErrorKey As String
Private mlngHighlightStart As Long
Private mlngHighlightEnd As Long

Private mbolErrorDisplayed As Boolean
Private mcolIgnoreRuleList As Collection

Private mbcSuggestion As cSuggestion
Private mfErrorForm As fErrorForm

Friend Property Get Suggestion() As cSuggestion
Set Suggestion = mbcSuggestion
End Property

Friend Property Get Result() As bcButtonPress
Result = mfErrorForm.Result
End Property

Friend Property Let Result(Source As bcButtonPress)
mfErrorForm.Result = Source
End Property

Friend Property Get ChangesMade() As Long
ChangesMade = mlngChangesMade
End Property

Friend Sub Hide()
mfErrorForm.Hide
End Sub

Public Sub LoadError(ByVal ErrorKey As String, ByVal HighlightStart As Long, ByVal HighlightEnd As Long, _
Optional ByVal AAA As String, Optional ByVal BBB As String, Optional ByVal CCC As String, Optional ByVal DDD As String)

1 If Not (ActiveDoc.Words.WordsTrim(HighlightEnd) = "") Then
If ActiveDoc.Words.SpacesAfterWord(HighlightEnd) = 0 And ActiveDoc.Words(HighlightEnd + 1) = "." Then HighlightEnd = HighlightEnd + 1
End If
Load the module-level variables that will be used later.
mbolErrorDisplayed = False
mlngHighlightStart = HighlightStart
mlngHighlightEnd = HighlightEnd
mstrErrorKey = ErrorKey
Load the error message and insert the AAA, BBB, CCC and DDDD information into it.
mstrErrorText = mbcErrorMessages.Item(ErrorKey).MessageText
If AAA <> "" Then mstrErrorText = Replace(mstrErrorText, "AAA", AAA)
If BBB <> "" Then mstrErrorText = Replace(mstrErrorText, "BBB", BBB)
If CCC <> "" Then mstrErrorText = Replace(mstrErrorText, "CCC", CCC)
If DDD <> "" Then mstrErrorText = Replace(mstrErrorText, "DDD", DDD)
Reset the suggestion
Call mbcSuggestion.Reset(Citation.CitationStart, Citation.CitationEnd)
End Sub

Public Function IgnoreError() As Boolean

Dim varErrorKey As Variant
Dim strTruncatedErrorKey As String
strTruncatedErrorKey = TruncatedErrorKey(mstrErrorKey)
For Each varErrorKey In mcolIgnoreRuleList
If varErrorKey = strTruncatedErrorKey Then
IgnoreError = True
Exit Function
End If
Next varErrorKey
End Function

Public Sub DisplaySuggestion()

With mfErrorForm
Call .AddSuggestion(mbcSuggestion.RTFNormal, mbcSuggestion.RTFHighlight)
If mbolErrorDisplayed = False Then Call DisplayError
End With
Reset the suggestion
Call mbcSuggestion.Reset(Citation.CitationStart, Citation.CitationEnd)
End Sub

Private Sub DisplayError()
Dim strReplacementText As String
Dim strOriginal As String
Dim strNormalRTF As String

```
Dim strHighlightRTF As String
Dim strRuleKey As String
```

```
'-- Select the citation in the original word document.
ActiveDoc.Words.Range(Citation.CitationStart, Citation.CitationEnd).Select
```

```
'-- Highlight the citation and display it.
strNormalRTF = ActiveDoc.Words.TextRTF(Citation.CitationStart, Citation.CitationEnd)
strHighlightRTF = HighlightRTF(ActiveDoc, strNormalRTF, mlngHighlightStart, mlngHighlightEnd)
Call mfErrorForm.AddCitation(strNormalRTF, strHighlightRTF)
```

```
'-- Substitute the "ORIGINAL" fields in the error message with the modified text.
If InStr(1, mstrErrorText, "ORIGINAL", vbBinaryCompare) Then
    strReplacementText = ActiveDoc.Words(mlngHighlightStart, mlngHighlightEnd)
    strReplacementText = FilterText(strReplacementText, "the", "the")
    mstrErrorText = Replace(mstrErrorText, "ORIGINAL", strReplacementText)
End If
```

```
'-- Substitute the "SUGGESTION" fields in the error message with the modified text.
If InStr(1, mstrErrorText, "SUGGESTION", vbBinaryCompare) Then
    strReplacementText = FilterText(mbcSuggestion.SuggestionText, "a", "an")
    mstrErrorText = Replace(mstrErrorText, "SUGGESTION", strReplacementText)
End If
```

```
'-- Convert the first character of the error message to upper case, and then display the error message.
mstrErrorText = Trim$(mstrErrorText)
mstrErrorText = UCase$(Left$(mstrErrorText, 1)) & Right$(mstrErrorText, Len(mstrErrorText) - 1)
mfErrorForm.txtErrorMessage.Text = mstrErrorText
```

```
'-- Display the rule associated with this rule.
strRuleKey = mbcErrorMessages.Item(mstrErrorKey).RuleKey
If strRuleKey = "T.1" Then
    If Not (Citation.Jurisdiction Is Nothing) Then
        If Not (Citation.Court Is Nothing) Then
            strRuleKey = "T.1: " & Citation.Jurisdiction & " " & Citation.Court.CourtGroup.FullName
        Else
            strRuleKey = ""
        End If
    End If
End If
If Not (strRuleKey = "") Then mfErrorForm.rtfRule = mbcBluebookRules.Item(strRuleKey).MessageText
mboErrorDisplayed = True
```

```
End Sub
```

```
Private Function FilterText(ByVal SourceText As String, ByVal ArticleOne As String, ByVal ArticleTwo As String) As String
```

```
SourceText = Trim$(SourceText)
If Len(Trim$(SourceText)) = 1 Then
    Select Case SourceText
        Case "("
            FilterText = ArticleTwo & " open parenthesis"
        Case ")"
            FilterText = ArticleOne & " close parenthesis"
        Case "["
            FilterText = ArticleTwo & " open bracket"
        Case "]"
            FilterText = ArticleOne & " close bracket"
        Case "."
            FilterText = ArticleOne & " period"
        Case ","
            FilterText = ArticleOne & " comma"
        Case "("
            FilterText = ArticleTwo & " open question mark"
        Case ";"
            FilterText = ArticleOne & " semicolon"
        Case ":"
            FilterText = ArticleOne & " colon"
        Case bcDBLQUOTE
            FilterText = ArticleOne & " quotation mark"
        Case "'"
            FilterText = ArticleTwo & " apostrophe"
        Case "-"
            FilterText = ArticleOne & " hyphen"
        Case Else
            FilterText = bcDBLQUOTE & SourceText & bcDBLQUOTE
    End Select
```

```
Else
    FilterText = bcDBLQUOTE & SourceText & bcDBLQUOTE
End If
```

```
End Function
```

```
Public Sub Activate()
```

```
If mbcSuggestion.IsDirty Then Call DisplaySuggestion
If mboErrorDisplayed = False Then Call DisplayError
```

```
With mfErrorForm
    Call .PrepareData
    .cmdCancel.Enabled = True
    .cmdIgnore.Enabled = True
    .cmdIgnoreRule.Enabled = True
    .cmdChange.Enabled = True
    .cmdChange.SetFocus
    .cmdChange.Enabled = False
End With
```

cErrorForm - 3

Call .MakeModal
End With

Call ProcessResults

End Sub

Private Sub ProcessResults()

Dim strTruncatedErrorKey

Select Case mfErrorForm.Result

Case Is = Change

ActiveDoc.Words.TextRTF(Citation.CitationStart, Citation.CitationEnd) = mfErrorForm.SelectedChangeRTF
mIngChangesMade = mIngChangesMade + 1

Case Is = IgnoreRule

mcolIgnoreRuleList.Add TruncatedErrorKey(mstrErrorKey)

End Select

End Sub

Private Sub Class_Initialize()

Set mbcBluebookRules = New cMessages

Call mbcBluebookRules.OpenFile("Bluebook Rules")

Set mbcErrorMessages = New cMessages

Call mbcErrorMessages.OpenFile("Error Messages")

Set mcolIgnoreRuleList = New Collection

Set mbcSuggestion = New cSuggestion

Set mfErrorForm = New fErrorForm

Load mfErrorForm

End Sub

Private Sub Class_Terminate()

Unload mfErrorForm

Set mfErrorForm = Nothing

End Sub

Private Function HighlightRTF(ByRef SourceDocument As cDocument, ByVal SourceRTF As String, ByVal FirstWord As Long, _
ByVal LastWord As Long) As String

Dim bcRichText As New cRichText

Dim lngFirstChar As Long

Dim lngLastChar As Long

Dim lngCharOffset As Long

If LastWord < FirstWord _

Or FirstWord = 0 _

Or LastWord = 0 Then

HighlightRTF = SourceRTF

Exit Function

End If

If SourceDocument Is ActiveDoc Then lngCharOffset = ActiveDoc.Words.FirstChar(Citation.CitationStart) - 1

lngFirstChar = SourceDocument.Words.FirstChar(FirstWord) - lngCharOffset

lngLastChar = SourceDocument.Words.LastChar(LastWord) - lngCharOffset

With bcRichText

.TextRTF = SourceRTF

.SelStart = lngFirstChar

.SelLength = lngLastChar - lngFirstChar

.SelColor = wdColorRed

HighlightRTF = .TextRTF

End With

End Function

Private Function TruncatedErrorKey(ErrorKey As String) As String

If IsNumeric(Right\$(mstrErrorKey, 1)) Then

TruncatedErrorKey = Trim\$(Left\$(mstrErrorKey, Len(ErrorKey) - 1))

Else

TruncatedErrorKey = mstrErrorKey

End If

End Function

cMessage - 1

13

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Public MessageName
Public MessageText
Public RuleKey

Public MessageName
Public MessageText
Public RuleKey

cMessages - 1

-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

```
Private mcolItems As Collection
Private mstrFileName As String
Private Const FILE_PATH As String = "C:\BlueCheck\Data\"
```

```
Friend Function Item(Index As Variant) As cMessage
    Set Item = mcolItems(Index)
End Function
```

```
Friend Sub Add(Item As cMessage, Optional Before, Optional After)
    mcolItems.Add Item, Item.MessageName, Before, After
End Sub
```

```
Friend Sub Remove(Index)
    mcolItems.Remove (Index)
End Sub
```

```
Friend Property Get FileName() As String
    FileName = mstrFileName
End Property
```

```
Public Function NewEnum() As IUnknown
    Set NewEnum = mcolItems.[_NewEnum]
End Function
```

```
Friend Sub OpenFile(FileName As String)
```

```
    Dim lngFileNumber As Long
    Dim bcMessage As cMessage
    Dim strMessageName As String
    Dim strMessageText As String
    Dim strRuleKey As String
```

```
    mstrFileName = FileName
    Set mcolItems = New Collection

    lngFileNumber = FreeFile
    Open FILE_PATH & mstrFileName & ".txt" For Input As lngFileNumber
    Do
        Line Input #lngFileNumber, strMessageName
        Line Input #lngFileNumber, strMessageText
        Line Input #lngFileNumber, strRuleKey

        Set bcMessage = New cMessage
        With bcMessage
            .MessageName = strMessageName
            .MessageText = strMessageText
            .RuleKey = strRuleKey
        End With
        mcolItems.Add bcMessage, bcMessage.MessageName
    Loop Until EOF(lngFileNumber)
    Close lngFileNumber
    mstrFileName = FileName
End Sub
```

```
Friend Sub SaveFile()
```

```
    Dim lngFileNumber As Long
    Dim bcMessage As cMessage
```

```
    lngFileNumber = FreeFile
    Open FILE_PATH & mstrFileName & ".txt" For Output As lngFileNumber

    For Each bcMessage In mcolItems
        With bcMessage
            Print #lngFileNumber, .MessageName
            Print #lngFileNumber, Replace(.MessageText, vbCrLf, "")
            Print #lngFileNumber, .RuleKey
        End With
    Next bcMessage
```

```
    Close lngFileNumber
    Call MsgBox("Save complete.", vbOKOnly, "Save complete")
```

```
End Sub
```

```
'-- Copyright 2001 by Robert L. Jacobson.
```

```
Option Explicit
```

```
Private mlngSuggestion() As Long
Private mbolHighlight() As Boolean
```

```
Private mbcSuggestion As cDocument
Private mlngFirstWord As Long
Private mlngLastWord As Long
```

```
Private mstrSuggestion As String
Private mbolIsDirty As Boolean
```

```
Private Const DELETED As Long = -100000000
```

```
Private Sub Class_Initialize()
```

```
    Set mbcSuggestion = New cDocument
    Call mbcSuggestion.Initialize
```

```
End Sub
```

```
Friend Property Get IsDirty() As Boolean
    IsDirty = mbolIsDirty
End Property
```

```
Friend Property Get SuggestionText() As String
    SuggestionText = mstrSuggestion
End Property
```

```
Friend Sub Reset(ByVal FirstWord As Long, ByVal LastWord As Long)
```

```
    Dim i As Long
    Dim lngOffset As Long
```

```
    mlngFirstWord = FirstWord
    mlngLastWord = LastWord
    mstrSuggestion = ""
```

```
    For i = mlngFirstWord To mlngLastWord
        ReDim mbcSuggestion(mlngFirstWord To mlngLastWord)
```

```
        mbcSuggestion.Range.FormattedText = ActiveDoc.Words.Range(mlngFirstWord, mlngLastWord).FormattedText
```

```
    mbcSuggestion.FormattedText = ActiveDoc.Words.Range(mlngFirstWord, mlngLastWord)
```

```
    lngOffset = Citation.CitationStart - 1
    For i = mlngFirstWord To mlngLastWord
        mlngSuggestion(i) = i - lngOffset
    Next i
```

```
    Call DebugPrintWords
    mbolIsDirty = False
```

```
End Sub
```

```
Friend Sub Change(ByVal FirstWord As Long, ByVal LastWord As Long, ByVal ReplacementText As String)
```

```
    Dim lngFirstWord As Long
    Dim lngLastWord As Long
    Dim lngFormerWordCount As Long
    Dim lngOffset As Long
    Dim i As Long
```

```
    lngFormerWordCount = mbcSuggestion.Words.Count
    lngFirstWord = mlngSuggestion(FirstWord)
    lngLastWord = mlngSuggestion(LastWord)
    If mstrSuggestion = "" Then mstrSuggestion = ReplacementText
```

```
    '-- If the word before the change start is a punctuation mark, and the change begins with the same punctuation mark, then
    ' delete the first punctuation mark when making the change.
```

```
    If mbcSuggestion.Words(lngFirstWord - 1) = Left$(Trim$(ReplacementText), 1) Then
        lngFirstWord = lngFirstWord - 1
    End If
```

```
    '-- If the word after the change end is a punctuation mark, and the change ends with the same punctuation mark, then
    ' delete the later punctuation mark when making the change.
```

```
    If mbcSuggestion.Words(lngLastWord + 1) = Right$(Trim$(ReplacementText), 1) Then
        lngLastWord = lngLastWord + 1
    End If
```

```
    '-- Change the suggestion.
```

```
    mbcSuggestion.Words.Highlight(lngFirstWord, lngLastWord) = True
    mbcSuggestion.Words(lngFirstWord, lngLastWord) = ReplacementText
```

```
    '-- Update the offset count.
```

```
    lngOffset = lngFormerWordCount - mbcSuggestion.Words.Count
    If lngOffset > 0 Then
```

```
        '-- The new suggestion is shorter than the old suggestion.
```

```
        For i = LastWord - lngOffset + 1 To LastWord
```

```
            mlngSuggestion(i) = DELETED
```

```
        Next i
```

```
        For i = LastWord + 1 To mlngLastWord
```

```
            mlngSuggestion(i) = mlngSuggestion(i) - lngOffset
```


End 540

cSuggestion - 3

```

    mbcSuggestion.Words.Highlight(lngInsertBeforeWord, lngOffset) = True

```

```

    '-- Update the offset count.

```

```

    For i = InsertBeforeWord To mlngLastWord
        mlngSuggestion(i) = mlngSuggestion(i) - lngOffset
    Next i

```

```

    Call DebugPrintWords
    mbcIsDirty = True

```

```

End Sub

```

```

Friend Sub Transpose(ByVal FirstWord As Long, ByVal LastWord As Long, ByVal InsertBeforeWord As Long)

```

```

    Dim strTextToTranspose As String
    Dim strSurroundingText As String

```

```

    strSurroundingText = ActiveDoc.Words(FirstWord - 1) & ActiveDoc.Words(LastWord + 1)
    Select Case strSurroundingText
        Case "("
            FirstWord = FirstWord - 1
            LastWord = LastWord + 1
    End Select

```

```

    strTextToTranspose = ActiveDoc.Words(FirstWord, LastWord)
    If mstrSuggestion = "" Then mstrSuggestion = strTextToTranspose
    Call Delete(FirstWord, LastWord)
    Call Insert(InsertBeforeWord, strTextToTranspose)

```

```

    mbcIsDirty = True

```

```

End Sub

```

```

Private Sub DebugPrintWords()

```

```

    Dim i As Long

```

```

    Debug.Print

```

```

    For i = mlngFirstWord To mlngLastWord

```

```

        If mlngSuggestion(i) <> DELETED Then

```

```

            Debug.Print i & ": ", ActiveDoc.Words(i), mbcSuggestion.Words(mlngSuggestion(i))

```

```

        Else

```

```

            Debug.Print i & ": ", ActiveDoc.Words(i)

```

```

        End If

```

```

    Next i

```

```

    Debug.Print "ActiveDoc : " & ActiveDoc.Words(mlngFirstWord, mlngLastWord)

```

```

    Debug.Print "Suggestion: " & mbcSuggestion.Words(mlngSuggestion(mlngFirstWord), mlngSuggestion(mlngLastWord))

```

```

End Sub

```

```

Public Property Get RTFNormal() As String

```

```

    RTFNormal = mbcSuggestion.RTF

```

```

End Property

```

```

Public Property Get RTFHighlight() As String

```

```

    ' TO DO: Make this actually display highlighted text

```

```

    strRTFHighlight = HighlightRTF(mbcSuggestion, strRTFNormal, lngTransposedStart, lngTransposedEnd)

```

```

    RTFHighlight = mbcSuggestion.RTF

```

```

End Property

```

```

'Private Function HighlightRTF(ByRef SourceDocument As cDocument, ByVal SourceRTF As String, ByVal FirstWord As Long, _
'ByVal LastWord As Long) As String

```

```

    Dim bcRichText As New cRichText

```

```

    Dim lngFirstChar As Long

```

```

    Dim lngLastChar As Long

```

```

    Dim lngCharOffset As Long

```

```

    If LastWord < FirstWord _

```

```

    Or FirstWord = 0 _

```

```

    Or LastWord = 0 Then

```

```

        HighlightRTF = SourceRTF

```

```

        Exit Function

```

```

    End If

```

```

    If SourceDocument Is ActiveDoc Then lngCharOffset = ActiveDoc.Words.FirstChar(Citation.CitationStart) - 1

```

```

    lngFirstChar = SourceDocument.Words.FirstChar(FirstWord)

```

```

    lngLastChar = SourceDocument.Words.LastChar(LastWord)

```

```

    With bcRichText

```

```

        .TextRTF = SourceRTF

```

```

        .SelStart = lngFirstChar - lngCharOffset

```

```

        .SelLength = lngLastChar - lngFirstChar - lngCharOffset

```

```

        .SelColor = wdColorRed

```

```

        HighlightRTF = .TextRTF

```

```

    End With

```

```

'End Function

```

fErrorForm - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Public Result As bcButtonPress

```
Private mstrCitationNormalRTF As String
Private mstrCitationHighlightRTF As String
Private mstrCitationText As String
Private mbolCitationIsHighlight As Boolean
```

```
Private mcolSuggestions As Collection
Private mstrAllSuggestionsHighlightRTF As String
Private mstrAllSuggestionsNormalRTF As String
Private mstrAllSuggestionsText As String
Private mbolSuggestionIsHighlight As Boolean
```

```
Private mstrSelectedChangeRTF As String
Private mlngActiveSelection As Long
```

```
Private mlngSuggestionStart() As Long
Private mlngSuggestionEnd() As Long
```

Private mbolModal As Boolean

```
Private Const bcCitationText = -1
Private Const bcSuggestionText = -2
Private Const SUGGESTION_NORMAL As Long = 795
Private Const SUGGESTION_EXPANDED As Long = 1750
```

Public Property Get SuggestionsCount() As Long

```
If mcolSuggestions Is Nothing Then
    SuggestionsCount = 0
Else
    SuggestionsCount = mcolSuggestions.Count
End If
```

End Property

```
Public Property Get SelectedChangeRTF() As String
    SelectedChangeRTF = mstrSelectedChangeRTF
End Property
```

Public Sub AddCitation(ByVal CitationNormalRTF As String, ByVal CitationHighlightRTF As String)

```
mstrCitationNormalRTF = CitationNormalRTF
mstrCitationHighlightRTF = CitationHighlightRTF

With rtfCitation
    .TextRTF = CitationHighlightRTF
    mstrCitationText = .Text
End With

mbolCitationIsHighlight = True
```

End Sub

Public Sub AddSuggestion(ByVal SuggestionNormalRTF As String, ByVal SuggestionHighlightRTF As String)

```
Dim lngSuggestionNumber

If mcolSuggestions Is Nothing Then Set mcolSuggestions = New Collection
mcolSuggestions.Add SuggestionNormalRTF

lngSuggestionNumber = SuggestionsCount

If lngSuggestionNumber = 1 Then
    lblSuggestion.Enabled = True
    rtfSuggestion.TextRTF = ""
End If
```

```
ReDim Preserve mlngSuggestionStart(1 To lngSuggestionNumber)
ReDim Preserve mlngSuggestionEnd(1 To lngSuggestionNumber)
```

With rtfSuggestion

```
If lngSuggestionNumber > 1 Then
    Call ExpandSuggestions
    .SelStart = Len(.Text)
    .SelText = vbCrLf
    .SelAlignment = rtfCenter
    .SelText = "--- or ---"
    .SelText = vbCrLf
    .SelAlignment = rtfLeft
End If
```

```
mlngSuggestionStart(lngSuggestionNumber) = .SelStart
.SelRTF = SuggestionHighlightRTF
mlngSuggestionEnd(lngSuggestionNumber) = .SelStart - 1
```

```
End With
mbolSuggestionIsHighlight = True
```

End Sub

Public Sub PrepareData()

Dim bcRichText As New cRichText

```

mstrAllSuggestionsText = rtfSuggestion.Text
mstrAllSuggestionsHighlightRTF = rtfSuggestion.TextRTF

If SuggestionsCount > 1 Then
    With bcRichText
        .TextRTF = mstrAllSuggestionsHighlightRTF
        .SelStart = 1
        .SelLength = Len(.Text)
        .SelColor = vbBlack
        mstrAllSuggestionsNormalRTF = .TextRTF
    End With
End If

End Sub

Public Sub MakeModal()

    If SuggestionsCount = 1 Then
        mlngActiveSelection = 1
        cmdChange.Enabled = True
        cmdChange.SetFocus
    End If

    Screen.MousePointer = vbDefault

    mbolModal = True
    Do
        DoEvents
    Loop Until mbolModal = False

End Sub

Public Sub MakeModeless()

    mbolModal = False

    Screen.MousePointer = vbHourglass
    Set mcolSuggestions = Nothing
    cmdIgnoreRule.Enabled = False
    cmdIgnore.Enabled = False
    cmdChange.Enabled = True
    cmdChange.SetFocus
    cmdChange.Enabled = False
    rtfSuggestion.Text = ""
    rtfSuggestion.Locked = True
    rtfSuggestion.Enabled = True
    rtfCitation.Text = ""
    rtfCitation.Locked = True
    rtfCitation.Enabled = True
    rtfRule.Text = ""
    rtfErrorMessage.Text = ""
    lblSuggestion.Enabled = False
    ContractSuggestions
    DoEvents

End Sub

Private Sub chkShowRule_Click()

    Dim lngOffset As Long

    lngOffset = rtfRule.Height
    If chkShowRule.Value = 0 Then
        rtfRule.Visible = False
        Me.Height = Me.Height - lngOffset
    Else
        rtfRule.Visible = True
        Me.Height = Me.Height + lngOffset
    End If

End Sub

Private Sub cmdCancel_Click()

    Result = Cancel
    Call MakeModeless
    Me.Hide

End Sub

Private Sub cmdChange_Click()

    Select Case mlngActiveSelection
        Case Is >= 1
            mstrSelectedChangeRTF = mcolSuggestions(mlngActiveSelection)
        Case Is = bcCitationText
            mstrSelectedChangeRTF = rtfCitation.TextRTF
        Case Is = bcSuggestionText
            mstrSelectedChangeRTF = rtfSuggestion.TextRTF
    End Select

    Result = Change
    Call MakeModeless

End Sub

Private Sub cmdIgnore_Click()

```

```

    Result = Ignore
    Call MakeModeless

```

```
End Sub
```

```
Private Sub cmdIgnoreRule_Click()
```

```

    Result = IgnoreRule
    Call MakeModeless

```

```
End Sub
```

```
Private Sub Form_Click()
```

```
    Me.SetFocus
```

```
End Sub
```

```
Private Sub lblCitation_Click()
```

```
    Me.SetFocus
```

```
End Sub
```

```
Private Sub lblSuggestion_Click()
```

```
    Me.SetFocus
```

```
End Sub
```

```
Private Sub Picture1_Click()
```

```
End Sub
```

```
Private Sub rtfCitation_Click()
```

```

    Dim lngSelectionStart As Long
    Dim lngSelectionLength As Long

```

```
If mbolCitationIsHighlight Then
```

```
    With rtfCitation
```

```

        lngSelectionStart = .SelStart
        lngSelectionLength = .SelLength
    
```

```

        rtfCitation.TextRTF = mstrCitationNormalRTF
        mbolCitationIsHighlight = False
        .Locked = False
    
```

```

        .SelStart = lngSelectionStart
        .SelLength = lngSelectionLength
    End With

```

```
    mbolCitationIsHighlight = False
```

```
End If
```

```
End Sub
```

```
Private Sub rtfCitation_LostFocus()
```

```
If rtfCitation.Text = mstrCitationText Then
```

```
    Call ResetCitation
```

```
If rtfSuggestion.Text = "" Then Call ResetSuggestion
```

```
End If
```

```
End Sub
```

```
Private Sub rtfCitation_KeyPress(KeyAscii As Integer)
```

```

    rtfSuggestion.Text = ""
    rtfSuggestion.Enabled = False
    lblSuggestion.Enabled = False

```

```
If KeyAscii = 13 Then
```

```

    KeyAscii = 0
    Call cmdChange_Click

```

```
End If
```

```

    mlngActiveSelection = bcCitationText
    cmdChange.Enabled = True
    cmdChange.SetFocus
    DoEvents

```

```
End Sub
```

```
Private Sub ResetCitation()
```

```
    With rtfCitation
```

```

        .Locked = True
        .Enabled = True
        .TextRTF = mstrCitationHighlightRTF
    End With

```

```
    mbolCitationIsHighlight = True
```

```
End Sub
```

```
Private Sub txtErrorMessage_Click()
```

```
    Me.SetFocus
```

End Sub

Private Sub rtfSuggestion_Click()

```
Dim lngSelectionStart As Long
Dim lngSelectionLength As Long
Dim i As Long
Dim lngCursorPosition As Long
```

With rtfSuggestion

If SuggestionsCount = 1 Then

If mbolSuggestionIsHighlight Then

```
lngSelectionStart = .SelStart
lngSelectionLength = .SelLength
```

```
.TextRTF = mcolSuggestions(1)
mbolSuggestionIsHighlight = False
.Locked = False
```

```
.SelStart = lngSelectionStart
.SelLength = lngSelectionLength
```

End If

ElseIf SuggestionsCount > 1 Then

```
lngCursorPosition = .SelStart
mngActiveSelection = 0
For i = 1 To SuggestionsCount
```

If lngCursorPosition >= mngSuggestionStart(i) And lngCursorPosition <= mngSuggestionEnd(i) Then

```
.SelStart = mngSuggestionStart(i)
.SelLength = mngSuggestionEnd(i) - mngSuggestionStart(i) + 1
cmdChange.Enabled = True
mngActiveSelection = i
Exit For
```

End If

Next i

If mngActiveSelection = 0 Then

```
cmdChange.Enabled = False
Me.SetFocus
```

End If

End If

End With

End Sub

Friend Sub rtfSuggestion_LostFocus()

```
If rtfSuggestion.Text = mstrAllSuggestionsText Then
Call ResetSuggestion
End If
```

End Sub

Private Sub rtfSuggestion_KeyPress(KeyAscii As Integer)

```
rtfCitation.Text = ""
rtfCitation.Enabled = False
```

```
If KeyAscii = 13 Then
KeyAscii = 0
Call cmdChange_Click
End If
```

```
mngActiveSelection = bcSuggestionText
cmdChange.Enabled = True
cmdChange.SetFocus
DoEvents
```

End Sub

Private Sub ResetSuggestion()

```
With rtfSuggestion
.Locked = True
.Enabled = True
.TextRTF = mstrAllSuggestionsHighlightRTF
End With
mbolSuggestionIsHighlight = True
```

End Sub

Private Sub Form_Load()

```
IsOnTop Me.hwnd, True
chkShowRule.Value = 0
```

```
Me.Show
Screen.MousePointer = vbHourglass
DoEvents
```

End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)

' If the user presses the close button from the control menu, stop the form from unloading and instead follow the shutdown
 ' procedure invoked from the Cancel button.

If UnloadMode = vbFormControlMenu Then

cmdCancel.Value = True

Cancel = True

Else

Cancel = False

End If

End Sub

Friend Sub ExpandSuggestions()

Dim ctlCurrent As Control

If Me.rtfSuggestion.Height <> SUGGESTION_EXPANDED Then

For Each ctlCurrent In Me.Controls

If TypeOf ctlCurrent Is Line Then

ctlCurrent.Y1 = ctlCurrent.Y1 + (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

ctlCurrent.Y2 = ctlCurrent.Y2 + (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

Else

If ctlCurrent.Top > Me.rtfSuggestion.Top Then ctlCurrent.Top = ctlCurrent.Top + (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

End If

Next ctlCurrent

Me.Height = Me.Height + (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

Me.rtfSuggestion.Height = Me.rtfSuggestion.Height + (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

End If

End Sub

Friend Sub ContractSuggestions()

Dim ctlCurrent As Control

If Me.rtfSuggestion.Height <> SUGGESTION_NORMAL Then

For Each ctlCurrent In Me.Controls

If TypeOf ctlCurrent Is Line Then

ctlCurrent.Y1 = ctlCurrent.Y1 - (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

ctlCurrent.Y2 = ctlCurrent.Y2 - (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

Else

If ctlCurrent.Top > Me.rtfSuggestion.Top Then ctlCurrent.Top = ctlCurrent.Top - (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

End If

Next ctlCurrent

Me.Height = Me.Height - (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

Me.rtfSuggestion.Height = Me.rtfSuggestion.Height - (SUGGESTION_EXPANDED - SUGGESTION_NORMAL)

End If

End Sub

fOptions - 1

'-- Copyright 2001 by Robert L. Jacobsen.

Option Explicit

Private mstrForCourt As String

```
Public Property Get ForCourt() As String
    ForCourt = mstrForCourt
End Property
```

Private Sub Form_Load()

Dim bcJurisdiction As cJurisdiction

IsOnTop Me.hwnd, True

For Each bcJurisdiction In Jurisdictions

```
    If Not (bcJurisdiction.FullName = "Federal") Then cmbState.AddItem (bcJurisdiction.FullName)
Next bcJurisdiction
```

optDocType(0).Value = False

optDocType(1).Value = False

optDocType(2).Value = False

End Sub

Private Sub cmdOK_Click()

If optDocType(0).Value = True And cmbState.Text <> "" Then

mstrForCourt = cmbState.Text

ElseIf optDocType(1).Value = True Then

mstrForCourt = "Federal"

ElseIf optDocType(2).Value = True Then

mstrForCourt = "None"

End If

Me.Visible = False

Debug.Print mstrForCourt

End Sub

Private Sub cmdCancel_Click()

Reset the display to the saved value.

Select Case mstrForCourt

Case "None"

optDocType(2).Value = True

Case "Federal"

optDocType(1).Value = True

Case ""

optDocType(0).Value = False

optDocType(1).Value = False

optDocType(2).Value = False

cmbState.Enabled = False

cmdOK.Enabled = False

Case Else

optDocType(0).Value = True

cmbState.Text = mstrForCourt

End Select

Me.Visible = False

End Sub

Private Sub optDocType_Click(Index As Integer)

Select Case Index

Case 0

'-- "For state" selected.

lblState.Enabled = True

cmbState.Enabled = True

cmbState.SetFocus

If cmbState.Text <> "" Then cmdOK.Enabled = True

Case 1, 2

'-- "For federal" or "other" selected.

lblState.Enabled = False

cmbState.Enabled = False

cmdOK.Enabled = True

End Select

End Sub

Private Sub cmbState_Click()

If optDocType(0).Value = True And cmbState.Text <> "" Then cmdOK.Enabled = True

End Sub



Dim x As Long

[illegible]

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Private mbolStartOfPhraseInitialized As Boolean

Private Const CASENAME_PLACEHOLDER As String = "_____ v. _____"

Public Sub CheckCaseName()

Dim lngVPos As Long
Dim strVText As String
Dim strOrigName As String

mbolStartOfPhraseInitialized = False

If Not Citation.CaseNameClause Is Nothing Then

If Not Citation.CaseNameClause.Text = CASENAME_PLACEHOLDER Then

Call CheckIncorrectVersusText
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckMultipleVersus
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckEtAlAndDBA
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckShouldBeExRel
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckShouldBeInRe
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckVersusFollowsProceduralPhrase
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckMultipleProceduralPhrases
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckInRePunctuation
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckExPartePunctuation
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckExRelPunctuation
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckFirstWordAbbreviated
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckThe
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckDescriptiveTerms
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckUsesStateOf
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckShouldNotUseState
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckShouldUseState
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckCityOf
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckPrepositionalPhraseOfLocation
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckOfAmerica
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckBusinessFirmDesignations
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckUnion
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckCIR
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckShouldAbbreviate
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckUnitedStates
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Else

Call CheckMissingCaseNameClause
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

End If

End If

End Sub

```
Private Sub CheckMissingCaseNameClause()
```

```
'-- If the citation doesn't have a case name clause, then suggest inserting a placeholder.

If Citation.CaseNameClause Is Nothing Then
    Call ErrorForm.LoadError("CaseName MissingCaseNameClause", 0, 0)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Insert(Citation.CitationStart, "_____ v. _____")
        Call ErrorForm.Activate
    End If
End If
```

```
End Sub
```

```
Private Sub CheckIncorrectVersusText()
```

```
'-- If the abbreviation for versus is not spelled as "v." with a trailing space, then suggest correcting it.
```

```
Dim i As Long
Dim strWordTrim As String
Dim lngVersusEnd As Long
```

```
For i = Citation.CaseNameClause.PartyName(1).Start To Citation.CaseNameClause.ClauseEnd
    strWordTrim = ActiveDoc.Words.WordsTrim(i)
```

```
If strWordTrim = "v" _
Or strWordTrim = "vs" _
Or strWordTrim = "versus" Then
    lngVersusEnd = ActiveDoc.Words.NextFullWord(i) - 1
```

```
'-- Is "versus" abbreviated incorrectly?
'>> Jones vs. Smith, 200 U.S. 500, 505 (1980)
```

```
If ActiveDoc.Words(i, lngVersusEnd) <> "v." Then
```

```
    Call ErrorForm.LoadError("CaseName IncorrectVersusText A", i, lngVersusEnd)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Change(i, lngVersusEnd, "v.")
        Call ErrorForm.Activate
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
    End If
```

```
'-- Is there no space after the abbreviation of "versus"?
'>> Jones v.Smith, 200 U.S. 500, 505 (1980)
```

```
ElseIf ActiveDoc.Words.SpacesAfterWord(lngVersusEnd) = 0 Then
```

```
    Call ErrorForm.LoadError("CaseName IncorrectVersusText B", i, lngVersusEnd)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Change(i, lngVersusEnd, "v.")
        Call ErrorForm.Activate
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
    End If
```

```
End If
End If
```

```
Next i
```

```
End Sub
```

```
Private Sub CheckMultipleVersus()
```

```
'-- If the case name has more than one abbreviations of "versus," then suggest deleting the subsequent versus.
```

```
Dim i As Long
Dim strWordTrim As String
Dim lngPriorComma As Long
Dim lngVersusEnd As Long
```

```
If Citation.CaseNameClause.PartiesCount > 1 Then
```

```
    For i = Citation.CaseNameClause.PartyName(2).Start To Citation.CaseNameClause.ClauseEnd
        strWordTrim = ActiveDoc.Words.WordsTrim(i)
```

```
If strWordTrim = "v" _
Or strWordTrim = "vs" _
Or strWordTrim = "versus" Then
```

```
'-- Search for a comma that falls between this versus and the first versus.
lngPriorComma = LocateWord(i - 1, Citation.CaseNameClause.VersusPosition, False, ",")
```

```
'-- If a prior comma was found, then suggest deleting it and all subsequent text.
'>> Jones v. Smith, Smith v. Jones, 200 U.S. 500, 505 (1980)
If lngPriorComma > 0 Then
```

```
    Call ErrorForm.LoadError("CaseName MultipleVersus 1", lngPriorComma + 1, Citation.CaseNameClause.ClauseEnd)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Delete(lngPriorComma, Citation.CaseNameClause.ClauseEnd)
        Call ErrorForm.Activate
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
    End If
```

```
'-- Otherwise, just ask the user to fix the error manually.
'>> Jones v. Smith Smith v. Jones, 200 U.S. 500, 505 (1980)
```

```
Else
    lngVersusEnd = ActiveDoc.Words.NextFullWord(i) - 1
```

```

Call ErrorForm.LoadError(CaseName MultipleVersus 2", i, lngVersusEnd)
If Not ErrorForm.IgnoreError Then
    Call ErrorForm.Activate
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
End If

```

```
End If
```

```
End If
```

```
Next i
```

```
End If
```

```
End Sub
```

```
Private Sub CheckEtAlAndDBA()
```

```

'-- Rule 10.2.1(a): If the case name contains the phrase "et al." or "d/b/a", then suggest deleting that phrase and the
'   remaining text in that party's name.
'>> Franklin et al. v. Amanant, 200 U.S. 500, 505 (1980)
'>> Franklin v. Amanant d/b/a The Man, Inc., 200 U.S. 500, 505 (1980)

```

```

Dim i As Long
Dim lngPhraseEnd As Long
Dim lngDeleteStart As Long
Dim strErrorKey As String

```

```
For i = Citation.CaseNameClause.PartyName(1).Start To Citation.CaseNameClause.CauseEnd
```

```

    lngPhraseEnd = IsStartOfPhrase(i, "etal", True)
    If lngPhraseEnd > 0 Then
        strErrorKey = "CaseName EtAlAndDBA A"
    Else
        lngPhraseEnd = IsStartOfPhrase(i, "dba", True)
        If lngPhraseEnd > 0 Then
            strErrorKey = "CaseName EtAlAndDBA B"
        End If
    End If
End If

```

```
If lngPhraseEnd > 0 Then
```

```

    If ActiveDoc.Words(i - 1) = "," Then
        lngDeleteStart = i - 1
    Else
        lngDeleteStart = i
    End If

```

```
Call ErrorForm.LoadError(strErrorKey, lngDeleteStart, Citation.CaseNameClause.PartyNameAtWord(i).End, ActiveDoc.Words(i, lngPhr
```

```

aseEnd))
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Delete(lngDeleteStart, Citation.CaseNameClause.PartyNameAtWord(i).End)
        Call ErrorForm.Activate
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
    End If

```

```
End If
```

```
Next i
```

```
End Sub
```

```
Private Sub CheckShouldBeExRel()
```

```

'-- Rule 10.2.1(b): If the case name contains the phrases "on... of," "for... of," or "by and through," then suggest changing the
'   phrase to "ex rel."

```

```

Dim i As Long
Dim lngOfLocation As Long
Dim strWordTrim As String
Dim lngPhraseEnd As Long

```

```
'-- Look for the phrase "on... of"
```

```
'>> Jones on Behalf of the Application of Smith v. Johnson, 200 U.S. 500, 505 (1980)
```

```
'!!! TO DO: Change this so the second party's name is checked as well.
```

```

For i = Citation.CaseNameClause.PartyName(1).Start To Citation.CaseNameClause.CauseEnd
    strWordTrim = ActiveDoc.Words.WordsTrim(i)
    If strWordTrim = "on" Or strWordTrim = "for" Then

```

```

        lngOfLocation = LocateWord(Citation.CaseNameClause.PartyName(1).End, i + 2, True, "of")
        If lngOfLocation > 0 Then

```

```

            If ActiveDoc.Words.WordsTrim(lngOfLocation + 1) = "the" Then
                lngPhraseEnd = lngOfLocation + 1
            Else
                lngPhraseEnd = lngOfLocation
            End If

```

```

            Call ErrorForm.LoadError("CaseName ShouldBeExRel", i, lngPhraseEnd)
            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Suggestion.Change(i, lngPhraseEnd, "ex rel.")
                Call ErrorForm.Activate
                If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
            End If

```

```
End If
```

```
End If
```

```
Next i
```

```

'-- Look for the phrases "by and through" and "for the use of"
'>> Jones by and through Smith v. Jones, 200 U.S. 500, 505 (1980)
For i = Citation.CaseNameClause.PartyName(1).Start To Citation.CaseNameClause.ClauseEnd
    lngOfLocation = IsStartOfPhrase(i, "byandthrough", False)
    If lngOfLocation = 0 Then lngOfLocation = IsStartOfPhrase(i, "fortheuseof", False)
    If lngOfLocation > 0 Then

        Call ErrorForm.LoadError("CaseName ShouldBeExRel", i, lngOfLocation)
        If Not ErrorForm.IgnoreError Then
            Call ErrorForm.Suggestion.Change(i, lngOfLocation, "ex rel.")
            Call ErrorForm.Activate
            If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
        End If
    End If
Next i

End Sub

Private Sub CheckShouldBeInRe()

'-- Rule 10.2.1(b): If the first word of the case name is "in," "petition," or "application" and that word is followed by "of,"
' then suggest changing the phrase to "In re".

'>> In the Matter of the Application of Smith, 200 U.S. 500, 505 (1980)
'>> Matter of the Application of Smith, 200 U.S. 500, 505 (1980)
'>> Petition of Smith, 200 U.S. 500, 505 (1980)

Dim i As Long
Dim lngOfLocation As Long
Dim strFirstWordTrim As String
Dim strSecondWordTrim As String

With Citation.CaseNameClause

    strFirstWordTrim = ActiveDoc.Words.WordsTrim(.PartyName(1).Start)
    strSecondWordTrim = ActiveDoc.Words.WordsTrim(.PartyName(1).Start + 1)

    If (strFirstWordTrim = "in" And strSecondWordTrim <> "re") _
    Or strFirstWordTrim = "petition" _
    Or strFirstWordTrim = "application" _
    Or strFirstWordTrim = "matter" Then

        lngOfLocation = LocateWord(.ClauseEnd, .PartyName(1).Start + 1, True, "of")
        If lngOfLocation > 0 Then

            Call ErrorForm.LoadError("CaseName ShouldBeInRe", .PartyName(1).Start, lngOfLocation)
            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Suggestion.Change(.PartyName(1).Start, lngOfLocation, "In re")
                Call ErrorForm.Activate
                If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
            End If
        End If
    End If
End With

End Sub

Private Sub CheckVersusFollowsProceduralPhrase()

'-- Rule 10.2.1(b): If a designation of "versus" follows "In re" or "Ex parte," then suggest deleting the procedural phrase.
'>> In re Jones v. Smith, 200 U.S. 500, 505 (1980)

Dim lngPhraseEnd As Long

With Citation.CaseNameClause

    If .VersusPosition > 0 Then
        lngPhraseEnd = IsStartOfPhrase(.PartyName(1).Start, "inre", True)
        If lngPhraseEnd = 0 Then lngPhraseEnd = IsStartOfPhrase(.PartyName(1).Start, "exparte", True)

        If lngPhraseEnd > 0 Then

            Call ErrorForm.LoadError("CaseName VersusFollowsProceduralPhrase", .PartyName(1).Start, lngPhraseEnd, ActiveDoc.Words(.PartyName(2).Start, .PartyName(2).End))
            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Suggestion.Delete(.PartyName(1).Start, lngPhraseEnd)
                Call ErrorForm.Activate
            End If
        End If
    End If
End With

End Sub

Private Sub CheckMultipleProceduralPhrases()

'-- Rule 10.2.1(b): If "ex rel." follows a designation of "In re" or "Ex parte," then suggest deleting "ex rel." and all of
' the text that follows it.
'>> In re Jones ex rel. Smith, 200 U.S. 500, 505 (1980)

Dim lngEndOfFirstPhrase As Long
Dim lngEndOfSecondPhrase As Long
Dim i As Long

```

```

With Citation.CaseNameClause
    lngEndOfFirstPhrase = IsStartOfPhrase(.PartyName(1).Start, "inre", True)
    If lngEndOfFirstPhrase = 0 Then lngEndOfFirstPhrase = IsStartOfPhrase(.PartyName(1).Start, "exparte", True)

    If lngEndOfFirstPhrase > 0 Then
        For i = lngEndOfFirstPhrase + 1 To .ClauseEnd
            lngEndOfSecondPhrase = IsStartOfPhrase(i, "exrel", True)

            If lngEndOfSecondPhrase > 0 Then
                Call ErrorForm.LoadError("CaseName MultipleProceduralPhrases", i, .ClauseEnd, ActiveDoc.Words(i, lngEndOfSecondPhrase),
                ActiveDoc.Words(.PartyName(1).Start, lngEndOfFirstPhrase))
                If Not ErrorForm.IgnoreError Then
                    Call ErrorForm.Suggestion.Delete(i, .ClauseEnd)
                    Call ErrorForm.Activate
                    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
                End If
            End If
        Next i
    End If

End With

End Sub

Private Sub CheckInRePunctuation()
    '-- Rule 10.2.1(b): If "In re" is incorrectly punctuated or capitalized, then suggest correcting it.
    '>> In Re. Jones, 200 U.S. 500, 505 (1980)

    Dim lngPhraseEnd As Long

    With Citation.CaseNameClause
        lngPhraseEnd = IsStartOfPhrase(.PartyName(1).Start, "inre", True)
        If lngPhraseEnd > 0 Then
            If ActiveDoc.Words(.PartyName(1).Start, lngPhraseEnd) <> "In re" Then
                Call ErrorForm.LoadError("CaseName InRePunctuation", .PartyName(1).Start, lngPhraseEnd)
                If Not ErrorForm.IgnoreError Then
                    Call ErrorForm.Suggestion.Change(.PartyName(1).Start, lngPhraseEnd, "In re")
                    Call ErrorForm.Activate
                    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
                End If
            End If
        End If
    End With

End Sub

Private Sub CheckExPartePunctuation()
    '-- Rule 10.2.1(b): If "Ex parte" is incorrectly punctuated or capitalized, then suggest correcting it.
    '>> Ex Parte Jones, 200 U.S. 500, 505 (1980)

    Dim lngPhraseEnd As Long

    With Citation.CaseNameClause
        lngPhraseEnd = IsStartOfPhrase(.PartyName(1).Start, "exparte", True)
        If lngPhraseEnd > 0 Then
            If ActiveDoc.Words(.PartyName(1).Start, lngPhraseEnd) <> "Ex parte" Then
                Call ErrorForm.LoadError("CaseName ExPartePunctuation", .PartyName(1).Start, lngPhraseEnd)
                If Not ErrorForm.IgnoreError Then
                    Call ErrorForm.Suggestion.Change(.PartyName(1).Start, lngPhraseEnd, "Ex parte")
                    Call ErrorForm.Activate
                    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
                End If
            End If
        End If
    End With

End Sub

End Sub

Private Sub CheckExRelPunctuation()
    '-- Rule 10.2.1(b): If "ex rel." is incorrectly punctuated or capitalized, then suggest correcting it.
    '>> Smith exrel. Jones, 200 U.S. 500, 505 (1980)

    Dim lngPhraseEnd As Long
    Dim i As Long

    With Citation.CaseNameClause
        For i = .PartyName(1).Start + 1 To .ClauseEnd - 1
            lngPhraseEnd = IsStartOfPhrase(i, "exrel", True)
            If lngPhraseEnd > 0 Then

```

```

If ActiveDoc.Words(i, lngPhraseEnd) <> "ex rel." Then
    Call ErrorForm.LoadError("CaseName ExRelPunctuation", i, lngPhraseEnd)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Change(i, lngPhraseEnd, "ex rel.")
        Call ErrorForm.Activate
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
    End If
End If

```

```

End If
End If
Next i

```

```
End With
```

```
End Sub
```

```
Private Sub CheckFirstWordAbbreviated()
```

```

'-- Rule 10.2.1(c): If the first word of a party name is an abbreviation (either an abbreviation from Table 6 or an
' unrecognized abbreviation) then suggest expanding it.
'>> Univ. of Minn. v. Smith, 200 U.S. 205, 210 (1980)

```

```

Dim lngParty As Long
Dim lngNameStart As Long
Dim lngNameEnd As Long

```

```
With Citation.CaseNameClause
```

```

For lngParty = 1 To .PartiesCount
    lngNameStart = .PartyName(lngParty).Start
    '-- "C.I.R." and "U.S." are handled elsewhere.
    If Not (IsStartOfPhrase(lngNameStart, "cir", False) > 0 Or IsStartOfPhrase(lngNameStart, "us", False) > 0) Then
        lngNameEnd = .PartyName(lngParty).End
        Call CheckImproperAbbreviation(lngNameStart, lngNameEnd, "CaseName FirstWordAbbreviated")
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
    End If
Next lngParty

```

```
End With
```

```
End Sub
```

```
End Sub
```

```
Private Sub CheckThe()
```

```

'-- Rule 10.2.1(d): If the first word of a party name is "The," then suggest deleting it unless it is part of the phrase
' "The King," "The Queen," or "The ... case(s)."
'>> The Univ. of Minn. v. Smith, 200 U.S. 205, 210 (1980)

```

```

Dim lngParty As Long
Dim lngNameStart As Long
Dim lngThePosition As Long

```

```
With Citation.CaseNameClause
```

```

For lngParty = 1 To .PartiesCount
    lngNameStart = .PartyName(lngParty).Start

    If ActiveDoc.Words.WordsTrim(lngNameStart) = "the" Then
        lngThePosition = lngNameStart

        '-- Is "The" not immediately followed by "King" or "Queen", and not followed by "Case" or "Cases"?
        If Not (ActiveDoc.Words.WordsTrim(lngThePosition + 1) = "queen" _
            Or ActiveDoc.Words.WordsTrim(lngThePosition + 1) = "king" _
            Or LocateWord(lngThePosition + 1, .ClauseEnd, True, "case", "cases") > 0) Then

            Call ErrorForm.LoadError("CaseName The", lngThePosition, lngThePosition)
            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Suggestion.Delete(lngThePosition, lngThePosition)
                Call ErrorForm.Activate
                If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
            End If
        End If
    End If
Next lngParty

```

```
End With
```

```

End If
Next lngParty

```

```
End With
```

```
End Sub
```

```
Private Sub CheckDescriptiveTerms()
```

```

'-- Rule 10.2.1(e): If the name of a party ends in a word that describes a party, like "administrator," "appellee," or
' "executor," then suggest deleting that word.
'>> Smith, Trustee v. Jones, 200 U.S. 500, 505 (1980)

```

```

Dim lngParty As Long
Dim lngNameEnd As Long
Dim strWordTrim As String
Dim lngDeleteStart As Long

```

```
For lngParty = 1 To Citation.CaseNameClause.PartiesCount
```

```

    lngNameEnd = Citation.CaseNameClause.PartyName(lngParty).End
    If lngNameEnd > Citation.CaseNameClause.PartyName(lngParty).Start Then

```

```
strWordTrim = ActiveDoc.Words.WordTrim(lngNameEnd)
If Len(strWordTrim) > 0 Then
```

```
'-- First, look for known descriptive terms.
If strWordTrim = "administrator" _
Or strWordTrim = "appellee" _
Or strWordTrim = "appellant" _
Or strWordTrim = "executor" _
Or strWordTrim = "executrix" _
Or strWordTrim = "licensee" _
Or strWordTrim = "licensor" _
Or strWordTrim = "trustee" Then
```

```
If ActiveDoc.Words(lngNameEnd - 1) = "," Then
    lngDeleteStart = lngNameEnd - 1
Else
    lngDeleteStart = lngNameEnd
End If
```

```
Call ErrorForm.LoadError("CaseName DescriptiveTerms A", lngNameEnd, lngNameEnd)
If Not ErrorForm.IgnoreError Then
    Call ErrorForm.Suggestion.Delete(lngDeleteStart, lngNameEnd)
    Call ErrorForm.Activate
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
End If
```

```
'-- Second, identify other descriptive terms by looking to their suffixes, but only if preceded by a comma.
ElseIf ActiveDoc.Words(lngNameEnd - 1) = "," Then
```

```
If Right$(strWordTrim, 2) = "or" _
Or Right$(strWordTrim, 3) = "ors" _
Or Right$(strWordTrim, 3) = "ant" _
Or Right$(strWordTrim, 4) = "ants" _
Or Right$(strWordTrim, 2) = "ee" _
Or Right$(strWordTrim, 3) = "ees" _
Or Right$(strWordTrim, 3) = "rix" _
Or Right$(strWordTrim, 5) = "rixes" Then
```

```
    lngDeleteStart = lngNameEnd - 1
```

```
    Call ErrorForm.LoadError("CaseName DescriptiveTerms B", lngNameEnd, lngNameEnd)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Delete(lngDeleteStart, lngNameEnd)
        Call ErrorForm.Activate
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
    End If
```

```
End If
```

```
End If
```

```
End If
```

```
End If
```

```
Next lngParty
```

```
End
```

```
Private Sub CheckUsesStateOf()
```

```
'-- Rule 10.2.1(f): If the case name contains "State of [State]", "Commonwealth of [State]" or "People of [State]",
and [state] is the jurisdiction of the citation, suggest deleting "of [state]." If, however, the citation has a different
jurisdiction, then suggest deleting "State/Commonwealth/People of."
```

```
Dim i As Long
Dim strPriorWordTrim As String
Dim lngOfPosition As Long
Dim bcMatchingJur As cJurisdiction
Dim lngJurEnd As Long
Dim lngDeleteStart As Long
```

```
For i = Citation.CaseNameClause.PartyName(1).Start + 1 To Citation.CaseNameClause.ClauseEnd
```

```
If ActiveDoc.Words.WordTrim(i) = "of" Then
```

```
    lngOfPosition = i
    strPriorWordTrim = ActiveDoc.Words.WordTrim(lngOfPosition - 1)
    If (strPriorWordTrim = "state" Or strPriorWordTrim = "commonwealth" Or strPriorWordTrim = "people") Then
```

```
        Set bcMatchingJur = JurisdictionAtWord(ActiveDoc.Words.NextFullWord(lngOfPosition), lngJurEnd)
        If Not (bcMatchingJur Is Nothing) Then
```

```
            '-- Is the jurisdiction that follows the "State/Commonwealth/People of" phrase the same as the state
            ' court for which this is being submitted (if any)?
```

```
If bcMatchingJur Is Citation.Jurisdiction Then
```

```
    '-- If so, suggest deleting "of [jurisdiction]"
```

```
'>> State of Minnesota v. Jones, 200 N.W.2d 200, 205 (Minn. Ct. App. 1990)
```

```
    Call ErrorForm.LoadError("CaseName UsesStateOf A", lngOfPosition, lngJurEnd, AddArticle(bcMatchingJur.FullName))
```

```
    If Not ErrorForm.IgnoreError Then
```

```
        Call ErrorForm.Suggestion.Delete(lngOfPosition, lngJurEnd)
```

```
        If ActiveDoc.Words.WordTrim(lngOfPosition - 2) = "the" Then Call ErrorForm.Suggestion.Delete(lngOfPosition - 2
```

```
, lngOfPosition - 2)
```

```
        Call ErrorForm.Activate
```

```
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
    End If
```

```
Else
```

```
    '-- Otherwise, suggest deleting "state/commonwealth/people of"
```



```

'>> Jones v. State of Minnesota, 200 N.W.2d 200, 205 (Minn. 1980)
If ActiveDoc.Words.WordsTrim(lngOfPosition - 2) = "the" Then
    lngDeleteStart = lngOfPosition - 2
Else
    lngDeleteStart = lngOfPosition - 1
End If

Call ErrorForm.LoadError("CaseName UsesStateOf B", lngDeleteStart, lngOfPosition, AddArticle(bcMatchingJur.FullName))

If Not ErrorForm.IgnoreError Then
    Call ErrorForm.Suggestion.Delete(lngDeleteStart, lngOfPosition)
    Call ErrorForm.Activate
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
End If

End If

End If

End If

Next i

End Sub

Private Sub CheckShouldNotUseState()

'-- Rule 10.2.1(f): If one party's name is simply "State," "Commonwealth," or "People," but the citation has a federal
' jurisdiction, then ask the user to change that party's name to the name of the state.
'>> State v. Jones, 200 U.S. 205, 210 (1980)

Dim lngParty As Long
Dim lngNameStart As Long
Dim strWordTrim As String

If Not (Citation.Jurisdiction Is Nothing) Then
    If Citation.Jurisdiction.FullName = "Federal" Then
        With Citation.CaseNameClause
            '-- Does the party name consist only of "state," "commonwealth" or "people"?
            For lngParty = 1 To .PartiesCount
                If .PartyName(lngParty).Start = .PartyName(lngParty).End Then
                    strWordTrim = ActiveDoc.Words.WordsTrim(.PartyName(lngParty).Start)
                    If strWordTrim = "state" Or strWordTrim = "commonwealth" Or strWordTrim = "people" Then
                        Call ErrorForm.LoadError("CaseName ShouldNotUseState", .PartyName(lngParty).Start, .PartyName(lngParty).End)
                        If Not ErrorForm.IgnoreError Then
                            Call ErrorForm.Activate
                            If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
                        End If
                    End If
                End If
            Next lngParty
        End With
    End If
End If

End Sub

Private Sub CheckShouldUseState()

'-- Rule 10.2.1(f): If one party's name consists only of the name of a state, and that state is the same as the citation's
' jurisdiction, then suggest changing the name of the state to "State," "Commonwealth," or "People."
'>> Minnesota v. Jones, 200 N.W.2d 200, 205 (Minn. Ct. App. 1990)

Dim lngParty As Long
Dim lngNameStart As Long
Dim strWordTrim As String
Dim lngJurEnd As Long
Dim bcMatchingJur As cJurisdiction

If Not (Citation.Jurisdiction Is Nothing) Then
    If Citation.Jurisdiction.FullName <> "Federal" Then
        With Citation.CaseNameClause
            '-- Does the party name consist only of the name or abbreviation of a state, and does that name match the
            ' citation's jurisdiction?
            For lngParty = 1 To .PartiesCount
                Set bcMatchingJur = JurisdictionAtWord(.PartyName(lngParty).Start, lngJurEnd)
                If bcMatchingJur Is Citation.Jurisdiction Then
                    If lngJurEnd = .PartyName(lngParty).End Then
                        Call ErrorForm.LoadError("CaseName ShouldUseState", .PartyName(lngParty).Start, .PartyName(lngParty).End, AddArticle(Citation.Jurisdiction.FullName))
                        If Not ErrorForm.IgnoreError Then
                            Call ErrorForm.Suggestion.Change(.PartyName(lngParty).Start, .PartyName(lngParty).End, "State")
                            Call ErrorForm.DisplaySuggestion
                            Call ErrorForm.Suggestion.Change(.PartyName(lngParty).Start, .PartyName(lngParty).End, "Commonwealth")
                            Call ErrorForm.DisplaySuggestion
                            Call ErrorForm.Suggestion.Change(.PartyName(lngParty).Start, .PartyName(lngParty).End, "People")
                            Call ErrorForm.Activate
                            If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
                        End If
                    End If
                End If
            Next lngParty
        End With
    End If
End If

```

```

        End If
    End If
Next lngParty

```

```

End With
End If
End If

```

```
End Sub
```

```
Private Sub CheckCityOf()
```

```

'-- Rule 10.2.1(f): If the name contains "City of," "Town of," "Township of," "Borough of," or "County of" and that phrase is
' not the first word of a party name, then suggest deleting the phrase.
'>> Mayor of the City of Minneapolis v. Jones, 200 U.S. 205, 210 (1980)

```

```

Dim i As Long
Dim lngParty As Long
Dim lngPhraseEnd As Long
Dim lngPhraseStart As Long

```

```
With Citation.CaseNameClause
```

```

    For lngParty = 1 To .PartiesCount
        For i = .PartyName(lngParty).Start + 1 To .PartyName(lngParty).End - 1

```

```

            lngPhraseEnd = IsStartOfPhrase(i, "cityof", False)
            If lngPhraseEnd = 0 Then lngPhraseEnd = IsStartOfPhrase(i, "townof", False)
            If lngPhraseEnd = 0 Then lngPhraseEnd = IsStartOfPhrase(i, "townof", False)
            If lngPhraseEnd = 0 Then lngPhraseEnd = IsStartOfPhrase(i, "townshipof", False)
            If lngPhraseEnd = 0 Then lngPhraseEnd = IsStartOfPhrase(i, "boroughof", False)
            If lngPhraseEnd = 0 Then lngPhraseEnd = IsStartOfPhrase(i, "countyof", False)

```

```
        If lngPhraseEnd > 0 Then
```

```

            If ActiveDoc.Words.WordsTrim(i - 1) = "the" Then
                lngPhraseStart = i - 1

```

```

            Else
                lngPhraseStart = i
            End If

```

```
            Call ErrorForm.LoadError("CaseName CityOf", lngPhraseStart, lngPhraseEnd)
```

```
            If Not ErrorForm.IgnoreError Then
```

```
                Call ErrorForm.Suggestion.Delete(lngPhraseStart, lngPhraseEnd)
```

```
                Call ErrorForm.Activate
```

```
                If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
            End If
```

```
        End If
```

```
    Next i
```

```
Next lngParty
```

```
End With
```

```
End Sub
```

```
Private Sub CheckPrepositionalPhraseOfLocation()
```

```

Rule 10.2.1(f): If a party's name contains "of," and the word "of" is not the first or second word in the party's name,
then suggest deleting it if it is a prepositional phrase of location.
>> Jones Corp. of California v. Smith, 200 U.S. 205, 210 (1980)

```

```

Dim i As Long
Dim lngParty As Long
Dim bcJurisdiction As cJurisdiction
Dim lngEndOfPhrase As Long
Dim strPriorWordTrim As String
Dim strNextWordTrim As String

```

```
With Citation.CaseNameClause
```

```

    For lngParty = 1 To .PartiesCount
        For i = .PartyName(lngParty).Start + 2 To .PartyName(lngParty).End - 1

```

```
        If ActiveDoc.Words.WordsTrim(i) = "of" Then
```

```
            '-- These exceptions are handled elsewhere.
```

```
            strPriorWordTrim = ActiveDoc.Words.WordsTrim(i - 1)
```

```
            strNextWordTrim = ActiveDoc.Words.WordsTrim(i + 1)
```

```
            If Not (strPriorWordTrim = "state" Or strPriorWordTrim = "commonwealth" Or strPriorWordTrim = "people") _
            And Not (strNextWordTrim = "america" Or strNextWordTrim = "am") Then
```

```
                Set bcJurisdiction = JurisdictionAtWord(i + 1, lngEndOfPhrase)
```

```
                If Not (bcJurisdiction Is Nothing) Then
```

```
                    Call ErrorForm.LoadError("CaseName PrepositionalPhraseOfLocation A", i, lngEndOfPhrase)
```

```
                    If Not ErrorForm.IgnoreError Then
```

```
                        Call ErrorForm.Suggestion.Delete(i, lngEndOfPhrase)
```

```
                        Call ErrorForm.Activate
```

```
                        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
                    End If
```

```
                Else
```

```
                    Call ErrorForm.LoadError("CaseName PrepositionalPhraseOfLocation B", i, .PartyName(lngParty).End)
```

```
                    If Not ErrorForm.IgnoreError Then
```

```
                        Call ErrorForm.Suggestion.Delete(i, .PartyName(lngParty).End)
```

```
                        Call ErrorForm.Activate
```

```
                        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
                    End If
```

```
        End If
```

mCheckCaseName - 10

```

        End If
    End If

    Next i
Next lngParty
End With

```

End Sub

Private Sub CheckOfAmerica()

```

'-- Rule 10.2.1(f): If the case name contains "United States of America," suggest deleting "of America."
'>> Smith v. United States of America, 200 U.S. 205, 210 (1980)

```

Dim i As Long

```

For i = Citation.CaseNameClause.ClauseStart To Citation.CaseNameClause.ClauseEnd
    If ActiveDoc.Words.WordsTrim(i, i + 1) = "unitedstates" Then
        If ActiveDoc.Words.WordsTrim(i + 2, i + 3) = "ofamerica" _
            Or ActiveDoc.Words.WordsTrim(i + 2, i + 3) = "ofam" Then

            Call ErrorForm.LoadError("CaseName OfAmerica", i + 2, i + 3)
            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Suggestion.Delete(i + 2, i + 3)
                Call ErrorForm.Activate
                If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
            End If
        End If
    End If
Next i

```

End Sub

Private Sub CheckBusinessFirmDesignations()

```

' Rule 10.2.1(h): If a party's name has more than one phrase that designates the party as being a business entity, then
' suggest deleting the later phrase.
' Smith Corp., Inc. v. Jones, 200 U.S. 200, 205 (1980)

```

```

Dim lngParty As Long
Dim i As Long
Dim j As Long
Dim lngPhraseEnd As Long
Dim lngSecondPhraseEnd As Long
Dim lngDeleteStart As Long

```

With Citation.CaseNameClause

```

    For lngParty = 1 To .PartiesCount
        For i = .PartyName(lngParty).Start + 1 To .PartyName(lngParty).End

            lngPhraseEnd = IsStartOfBusinessDesignation(i)
            If lngPhraseEnd > 0 Then

                For j = lngPhraseEnd + 1 To .PartyName(lngParty).End
                    lngSecondPhraseEnd = IsStartOfBusinessDesignation(j)
                    If lngSecondPhraseEnd > 0 Then

                        If ActiveDoc.Words(j - 1) = "," Then
                            lngDeleteStart = j - 1
                        Else
                            lngDeleteStart = j
                        End If

                        Call ErrorForm.LoadError("CaseName CheckBusinessFirmDesignation", j, lngSecondPhraseEnd, ActiveDoc.Words(i, lng
PhraseEnd))

                        If Not ErrorForm.IgnoreError Then
                            Call ErrorForm.Suggestion.Delete(lngDeleteStart, lngSecondPhraseEnd)
                            Call ErrorForm.Activate
                            Exit For
                        End If
                    End If
                Next j
            End If
        Next i
    Next lngParty
End With

```

End Sub

Private Function IsStartOfBusinessDesignation(WordNumber As Long) As Long

```

IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "inc", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "incorporated", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "inc", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "limited", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "ltd", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "brothers", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "bros", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "association", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "assn", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "assoc", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "rr", True)

```

mCheckCaseName - 11

```
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "ry", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "corp", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "co", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "na", True)
If IsStartOfBusinessDesignation = 0 Then IsStartOfBusinessDesignation = IsStartOfPhrase(WordNumber, "fsb", True)
```

End Function

Private Sub CheckUnion()

```
'-- Rule 10.2.1(i): If a party's name has a word indicating a union designation, then suggest that the user consult
' Rule 10.2.1(i) to ensure that it complies with that rule.
```

```
Dim lngParty As Long
Dim i As Long
Dim strWordTrim As String
```

With Citation.CaseNameClause

```
For lngParty = 1 To .PartiesCount
    For i = .PartyName(lngParty).Start To .PartyName(lngParty).End
        strWordTrim = ActiveDoc.Words.WordsTrim(i)
        If strWordTrim = "local" _
        Or strWordTrim = "union" _
        Or strWordTrim = "international" _
        Or strWordTrim = "brotherhood" Then
```

```
            Call ErrorForm.LoadError("CaseName CheckUnion", .PartyName(lngParty).Start, .PartyName(lngParty).End)
            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Activate
            End If
```

```
        End If
    Next i
Next lngParty
End With
```

End Sub

Private Sub CheckCIR()

```
'-- Rule 10.2.1(j): If a party's name starts with "C.I.R." or "Commissioner of Internal Revenue," then suggest changing the
' phrase to "Commissioner."
' CIR v. Jones, 200 U.S. 205, 210 (1980)
```

```
Dim lngParty As Long
Dim lngPartyStart As Long
Dim lngPhraseEnd As Long
```

With Citation.CaseNameClause

```
For lngParty = 1 To .PartiesCount
    lngPartyStart = .PartyName(lngParty).Start
    lngPhraseEnd = IsStartOfPhrase(lngPartyStart, "cir", True)
    If lngPhraseEnd = 0 Then lngPhraseEnd = IsStartOfPhrase(lngPartyStart, "commissionerofinternalrevenue", True)

    If lngPhraseEnd > 0 Then
        Call ErrorForm.LoadError("CaseName CIR", lngPartyStart, lngPhraseEnd)
        If Not ErrorForm.IgnoreError Then
            Call ErrorForm.Suggestion.Change(lngPartyStart, lngPhraseEnd, "Commissioner")
            Call ErrorForm.Activate
        End If
    End If
Next lngParty
End With
```

End Sub

Private Sub CheckShouldAbbreviate()

```
'-- Rule 10.2.2: If one party's name contains a full word from Table 6 or 10, or the full name of a jurisdiction,
' and the word is not the first word of the name of the party, then suggest replacing the word with its abbreviation.
'>> University of Minnesota v. Jones, 200 U.S. 205, 210 (1980).
```

```
Dim lngParty As Long
Dim i As Long
Dim strMatchingAbrv As String
Dim lngNameEnd As Long
```

With Citation.CaseNameClause

```
For lngParty = 1 To .PartiesCount
    For i = .PartyName(lngParty).Start + 1 To .PartyName(lngParty).End

        strMatchingAbrv = AbrvForFullName(i, .PartyName(lngParty).End, lngNameEnd)
        If strMatchingAbrv <> "" And strMatchingAbrv <> "U.S." Then

            Call ErrorForm.LoadError("CaseName ShouldAbbreviate", i, lngNameEnd)
            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Suggestion.Change(i, lngNameEnd, strMatchingAbrv)
                Call ErrorForm.Activate
            End If

        End If

    Next i
Next lngParty
```

```

End With

End Sub

Private Sub CheckUnitedStates()

'-- Rule 10.2.2: If "United States" is abbreviated, suggest changing it to "United States."
'>> JonesCo of U.S. v. Smith, 200 U.S. 205, 210 (1980)

Dim lngParty As Long
Dim i As Long
Dim lngPhraseEnd As Long

With Citation.CaseNameClause

For lngParty = 1 To .PartiesCount
For i = .PartyName(lngParty).Start To .PartyName(lngParty).End
lngPhraseEnd = IsStartOfPhrase(i, "us", True)
If lngPhraseEnd > 0 Then

Call ErrorForm.LoadError("CaseName UnitedStates", i, lngPhraseEnd)
If Not ErrorForm.IgnoreError Then
Call ErrorForm.Suggestion.Change(i, lngPhraseEnd, "United States")
Call ErrorForm.Activate
End If

End If

Next i
Next lngParty

End With

End Sub

Private Sub CheckAbbreviation()

Dim i As Long
Dim abrMatch As cAbbreviation
Dim bolKnownAbrv As Boolean
Dim lngPhraseEnd As Long

Citation.Words.IsAbbreviation(WordNumber) Then

' *****
' If the case name contains an abbreviation from Table 6 or Table 10 that is
' punctuated incorrectly, suggest replacing it with the correct punctuation.
' *****

bolKnownAbrv = False
Set abrMatch = gTable6.MatchAbrv(WordNumber, , lngPhraseEnd)
If Not (abrMatch Is Nothing) Then
If Not Citation.Words(WordNumber, lngPhraseEnd) = abrMatch.FullNameTrim Then

bolKnownAbrv = True
If Not Citation.Words.IsMatch(Exact, abrMatch.AbrvName, WordNumber) Then
Suggestion.FormattedText = Citation.FormattedText
Call Suggestion.ReplaceWords(WordNumber, lngPhraseEnd, abrMatch.AbrvName)
Call mDisplayError.Change("T.6 Misspelled Abrv", WordNumber, lngPhraseEnd)
If gbolStartOver = True Then Exit Sub
End If

End If

End If

Set abrMatch = gTable10.MatchAbrv(WordNumber, , lngPhraseEnd)
If Not (abrMatch Is Nothing) Then
If Not Citation.Words(WordNumber, lngPhraseEnd) = abrMatch.FullNameTrim Then

bolKnownAbrv = True
If Not Citation.Words.IsMatch(Exact, abrMatch.AbrvName, WordNumber) Then
Suggestion.FormattedText = Citation.FormattedText
Call Suggestion.ReplaceWords(WordNumber, lngPhraseEnd, abrMatch.AbrvName)
Call mDisplayError.Change("T.10 Misspelled Abrv", WordNumber, lngPhraseEnd)
If gbolStartOver = True Then Exit Sub
End If

End If

End If

' *****
' If the case name contains an unknown abbreviation, call it to the user's
' attention.
' *****

If bolKnownAbrv = False Then

For i = WordNumber - 1 To WordNumber - 6 Step -1
Set abrMatch = gAllAbbreviations.MatchAbrv(i, , lngPhraseEnd)
If Not (abrMatch Is Nothing) Then
If lngPhraseEnd >= WordNumber Then bolKnownAbrv = True
Exit For
End If
Next i

```

mCheckCaseName - 13

```

    If bolKnownAbrv = False Then
        Call mDisplayError(WordNumber, 10.2.2 Unknown Abrv", WordNumber, WordNumber)
        If gbolStartOver = True Then Exit Sub
    End If

End If

End If

End Sub

```

```
Private Sub CheckImproperAbbreviation(WordNumber As Long, EndOfPartyName As Long, ErrorKey As String)
```

```

    Dim colFullNames As Collection
    Dim lngEndOfAbrv As Long
    Dim varFullName As Variant
    Dim strErrorKey As String

    '-- Does a party's name begin with a recognized abbreviation?
    Set colFullNames = FullNamesForAbrvs(WordNumber, EndOfPartyName, lngEndOfAbrv)
    If colFullNames.Count > 0 Then

        If colFullNames.Count = 1 Then
            strErrorKey = ErrorKey & " 2"
        Else
            strErrorKey = ErrorKey & " 3"
        End If

        Call ErrorForm.LoadError(strErrorKey, WordNumber, lngEndOfAbrv)
        If Not ErrorForm.IgnoreError Then
            For Each varFullName In colFullNames
                Call ErrorForm.Suggestion.Change(WordNumber, lngEndOfAbrv, varFullName)
                Call ErrorForm.DisplaySuggestion
            Next varFullName
            Call ErrorForm.Activate
            Exit Sub
        End If

    End If

    Does a party's name begin with a phrase that looks like an abbreviation, but wasn't identified above?
    lngEndOfAbrv = IsApparentAbrv(WordNumber, EndOfPartyName)
    If lngEndOfAbrv > 0 Then

        Call ErrorForm.LoadError(ErrorKey & " 1", WordNumber, lngEndOfAbrv)
        If Not ErrorForm.IgnoreError Then
            Call ErrorForm.Activate
            Exit Sub
        End If

    End If

End Sub

```

```
Private Function IsStartOfPhrase(FirstWord As Long, PhraseTrim As String, IncludeTrailingPeriod As Boolean) As Long
```

```

    Dim lngFirstWord As Long
    Dim lngLastWord As Long
    Dim lngPhraseLength As Long
    Dim lngCaseClauseEnd As Long
    Static strPhrases() As String

    If ActiveDoc.Words.WordsTrim(FirstWord) = "" Then Exit Function

    lngCaseClauseEnd = Citation.CaseNameClause.ClauseEnd
    If Not mbolStartOfPhraseInitialized Then
        ReDim strPhrases(Citation.CaseNameClause.ClauseStart To lngCaseClauseEnd, 1 To 10)
        For lngFirstWord = Citation.CaseNameClause.PartyName(1).Start To lngCaseClauseEnd

            For lngPhraseLength = 1 To 10
                lngLastWord = lngFirstWord + lngPhraseLength - 1
                If lngLastWord > lngCaseClauseEnd Then Exit For
                strPhrases(lngFirstWord, lngPhraseLength) = ActiveDoc.Words.WordsTrim(lngFirstWord, lngLastWord)
            Next lngPhraseLength

            Next lngFirstWord
            mbolStartOfPhraseInitialized = True
        End If

    For lngPhraseLength = 1 To 10
        If strPhrases(FirstWord, lngPhraseLength) = PhraseTrim Then
            IsStartOfPhrase = FirstWord + lngPhraseLength - 1
            If IncludeTrailingPeriod = True Then
                If ActiveDoc.Words(IsStartOfPhrase + 1) = "." Then IsStartOfPhrase = IsStartOfPhrase + 1
            End If
            Exit For
        End If
    Next lngPhraseLength

End Function

```

```
End Function
```

```
Private Function IsApparentAbrv(WordNumber As Long, EndOfPartyName As Long) As Long
```

```

    Dim strWord As String
    Dim strNextWord As String

```

```

If ActiveDoc.Words.WordsTrim(WordNumber) = "" Then
    strWord = ActiveDoc.Words(WordNumber)
    strNextWord = ActiveDoc.Words(WordNumber + 1)
    If ActiveDoc.Words.SpacesAfterWord(WordNumber) = 0 And ActiveDoc.Words(WordNumber + 1) = "." Then
        IsApparentAbrv = WordNumber
        Do While ActiveDoc.Words.SpacesAfterWord(IsApparentAbrv) = 0 And IsApparentAbrv <= EndOfPartyName
            IsApparentAbrv = IsApparentAbrv + 1
        Loop
    ElseIf strWord = UCase(strWord) And Len(strWord) > 1 Then
        IsApparentAbrv = WordNumber
    ElseIf strNextWord = "" Then
        If ActiveDoc.Words.SpacesAfterWord(WordNumber + 1) = 0 And Not (ActiveDoc.Words(WordNumber + 2) = "s") Then
            IsApparentAbrv = WordNumber + 2
        End If
    End If
End If

```

End Function

Private Function FullNamesForAbrvs(WordNumber As Long, EndOfPartyName As Long, Optional EndOfAbrv As Long) As Collection

```

Dim lngLastWord As Long
Dim strPhraseTrim As String
Dim lngEndOfMatch As Long
Dim lngLongestMatch As Long
Dim bcAbrv As cAbbreviation
Dim bcJurisdiction As cJurisdiction

```

Set FullNamesForAbrvs = New Collection

If Not (ActiveDoc.Words.WordsTrim(WordNumber) = "") Then

For lngLastWord = WordNumber + 9 To WordNumber Step -1

If lngLastWord <= EndOfPartyName Then

If ActiveDoc.Words.WordsTrim(lngLastWord) <> "" Then

strPhraseTrim = ActiveDoc.Words.WordsTrim(WordNumber, lngLastWord)

For Each bcAbrv In AbrvsAll

If strPhraseTrim = bcAbrv.AbrvNameTrim Then

If bcAbrv.AbrvName <> bcAbrv.FullName Then

FullNamesForAbrvs.Add bcAbrv.FullName

EndOfAbrv = lngLastWord

End If

End If

Next bcAbrv

For Each bcJurisdiction In Jurisdictions

If strPhraseTrim = bcJurisdiction.AbrvNameTrim Then

If bcJurisdiction.AbrvName <> bcJurisdiction.FullName Then

FullNamesForAbrvs.Add bcJurisdiction.FullName

EndOfAbrv = lngLastWord

End If

End If

Next bcJurisdiction

If FullNamesForAbrvs.Count > 0 Then

If ActiveDoc.Words(EndOfAbrv + 1) = "." Then EndOfAbrv = EndOfAbrv + 1

Exit Function

End If

End If

End If

Next lngLastWord

End If

End Function

Private Function AbrvForFullName(WordNumber As Long, EndOfPartyName As Long, Optional EndOfFullName As Long) As String

```

Dim lngLastWord As Long
Dim bcAbrv As cAbbreviation
Dim lngLongestMatch As Long
Dim strPhraseTrim As String
Dim bcJurisdiction As cJurisdiction

```

If Not (ActiveDoc.Words.WordsTrim(WordNumber) = "") Then

For lngLastWord = WordNumber + 9 To WordNumber Step -1

If lngLastWord <= EndOfPartyName And ActiveDoc.Words.WordsTrim(lngLastWord) <> "" Then

strPhraseTrim = ActiveDoc.Words.WordsTrim(WordNumber, lngLastWord)

For Each bcAbrv In AbrvsAll

If bcAbrv.HasAbbreviation Then

If strPhraseTrim = bcAbrv.FullNameTrim Then

If bcAbrv.AbrvName <> bcAbrv.FullName Then

AbrvForFullName = bcAbrv.AbrvName

EndOfFullName = lngLastWord

Exit Function

End If

End If

End If

Next bcAbrv

For Each bcJurisdiction In Jurisdictions

mCheckCaseName - 15

```

        If strPhraseTrim = bcJurisdiction.FullNameTrim Then
            If bcJurisdiction.AbrvName <> bcJurisdiction.FullName Then
                AbrvFullName = bcJurisdiction.AbrvName
                EndOfPhraseName = lngLastWord
                Exit Function
            End If
        End If
    Next bcJurisdiction

```

```

    End If
Next lngLastWord

```

```

End If

```

```

End Function

```

```

Private Function LocateWord(SearchFrom As Long, SearchTo As Long, SearchAsTrim As Boolean, ParamArray SearchFor()) As Long

```

```

    Dim lngStep As Long
    Dim lngWord As Long
    Dim strWord As String
    Dim varSearchForWord As Variant

```

```

    If SearchTo > SearchFrom Then
        lngStep = 1
    Else
        lngStep = -1
    End If

```

```

    For lngWord = SearchFrom To SearchTo Step lngStep
        If SearchAsTrim Then
            strWord = ActiveDoc.Words.WordsTrim(lngWord)
        Else
            strWord = ActiveDoc.Words.Words(lngWord)
        End If
        For Each varSearchForWord In SearchFor
            If strWord = varSearchForWord Then
                LocateWord = lngWord
                Exit Function
            End If
        Next varSearchForWord
    Next lngWord

```

```

End Function

```

```

Private Function JurisdictionAtWord(FirstWord As Long, Optional EndOfPhrase As Long) As cJurisdiction

```

```

    Dim i As Long
    Dim lngClauseEnd As Long

    If ActiveDoc.Words.WordsTrim(FirstWord) = "" Then Exit Function

    lngClauseEnd = Citation.CaseNameClause.ClauseEnd
    For i = FirstWord + 1 To FirstWord Step -1
        If i <= lngClauseEnd Then
            If Not ActiveDoc.Words.WordsTrim(i) = "" Then
                Set JurisdictionAtWord = Jurisdictions.Item(ActiveDoc.Words.WordsTrim(FirstWord, i), True, False, False)

                If Not JurisdictionAtWord Is Nothing Then
                    If ActiveDoc.Words(i + 1) = "." Then
                        EndOfPhrase = i + 1
                    Else
                        EndOfPhrase = i
                    End If
                End If
            End If
        End If
    Next i

```

```

End Function

```


'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Public Sub CheckClauseOrder()

```

    Call CheckReporterClausePosition
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

    Call CheckJurisdictionPosition
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

    Call CheckCourtPosition
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

    Call CheckDatePosition
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

```

End Sub

Private Sub CheckReporterClausePosition()

'-- If the first reporter clause does not follow the case name clause, then suggest moving all contiguous reporter clauses to follow the case name clause.

```

Dim bcFirstReporterClause As IClause
Dim bcLastReporterClause As IClause
Dim bcCaseNameClause As IClause
Dim lngMoveToPosition As Long

```

```

If Citation.ReporterClauses.Count > 0 Then
    Set bcFirstReporterClause = Citation.ReporterClauses(1)

```

```

    If Not (Citation.CaseNameClause Is Nothing) Then
        Set bcCaseNameClause = Citation.CaseNameClause

```

```

        If Not (bcCaseNameClause.NextClause Is bcFirstReporterClause) Then

```

```

            Set bcLastReporterClause = Citation.ReporterClauses(1)
            Do While Not (bcLastReporterClause.NextClause Is Nothing)
                If TypeOf bcLastReporterClause.NextClause Is cClauseReporter Then
                    Set bcLastReporterClause = bcLastReporterClause.NextClause
                Else
                    Exit Do
                End If
            Loop

```

```

            If ActiveDoc.Words(bcCaseNameClause.ClauseEnd + 1) = "," Then
                lngMoveToPosition = bcCaseNameClause.ClauseEnd + 2
            Else
                lngMoveToPosition = bcCaseNameClause.ClauseEnd + 1
            End If

```

```

            Call ErrorForm.LoadError("ClauseOrder CheckReporterPosition", bcFirstReporterClause.ClauseStart, bcLastReporterClause.ClauseEnd, lngMoveToPosition)

```

```

            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Suggestion.Transpose(bcFirstReporterClause.ClauseStart, bcLastReporterClause.ClauseEnd, lngMoveToPosition)

```

```

                Call ErrorForm.Activate
            End If

```

```

        End If

```

```

    End If

```

```

End Sub

```

End Sub

Private Sub CheckJurisdictionPosition()

'-- If the jurisdiction is not the first word in the court date clause, suggest moving it to the correct position.

```

Dim bcExpectedPriorClause As IClause
Dim bolMoveCourt As Boolean
Dim bolMoveDate As Boolean
Dim lngMoveToPosition As Long

```

```

If Not (Citation.JurisdictionClause Is Nothing) Then

```

'-- Determine which clause should precede the jurisdiction clause (usually the last reporter clause, but, if there is no reporter clause, then the case name clause.)

```

If Citation.ReporterClauses.Count > 0 Then
    Set bcExpectedPriorClause = Citation.ReporterClauses(Citation.ReporterClauses.Count)
ElseIf Not (Citation.CaseNameClause Is Nothing) Then
    Set bcExpectedPriorClause = Citation.CaseNameClause
End If

```

'-- Check whether the jurisdiction clause is in the correct position.

```

If Not (bcExpectedPriorClause.NextClause Is Citation.JurisdictionClause) Then
    If Not (bcExpectedPriorClause.NextClause Is Citation.CourtClause) Then

```

'-- Does the court clause follow the jurisdiction clause?

```

If Citation.JurisdictionClause.NextClause Is Citation.CourtClause Then
    If Not InStr(ActiveDoc.Words(Citation.JurisdictionClause.JurEnd + 1, Citation.CourtClause.CourtStart - 1), ",") Then
        bolMoveCourt = True
    End If
End If

```

```
Dim bcExpectedPriorClause As IClause
Dim bolMoveDate As Boolean
Dim lngMoveToPosition As Long
```

```
Call ErrorForm.LoadError("Dec CourtPos A", Citation.CourtClause.CourtStart, Citation.DateClause.DateEnd)
If Not ErrorForm.IgnoreError Then
```

mCheckClauseOrder - 3

```

    Call ErrorForm.Suggestion.Transpose(Citation.CourtClause.CourtStart, Citation.DateClause.DateEnd, lngMoveToPosition)
)
    Call ErrorForm.Activate
End If

Else
    '>> Jones v. Smith, 2d Cir., 123 F.2d 456 (1980)

    Call ErrorForm.LoadError("Dec CourtPos B", Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Transpose(Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd, lngMoveToPosition)

        Call ErrorForm.Activate
    End If

End If

End If

End If

End Sub

Private Sub CheckDatePosition()

    '-- If the date clause does not follow the court clause, jurisdiction clause or last reporter, then suggest moving it to
    ' the correct position.
    '>> Jones v. Smith, 1980, 123 F.2d 456 (2d Cir.)

    Dim bcExpectedPriorClause As IClause
    Dim lngMoveToPosition As Long

    If Not (Citation.DateClause Is Nothing) Then

        '-- Determine which clause should precede the jurisdiction clause (usually the last reporter clause, but, if there is no
        ' reporter clause, then the case name clause.)

        If Not (Citation.CourtClause Is Nothing) Then
            Set bcExpectedPriorClause = Citation.CourtClause
        ElseIf Not (Citation.JurisdictionClause Is Nothing) Then
            Set bcExpectedPriorClause = Citation.JurisdictionClause
        ElseIf Citation.ReporterClauses.Count > 0 Then
            Set bcExpectedPriorClause = Citation.ReporterClauses(Citation.ReporterClauses.Count)
        End If

        -- Check whether the jurisdiction clause is in the correct position.
        If Not (bcExpectedPriorClause Is Nothing) Then
            If Not (bcExpectedPriorClause.NextClause Is Citation.DateClause) Then

                lngMoveToPosition = bcExpectedPriorClause.ClauseEnd + 1
                If ActiveDoc.Words(lngMoveToPosition) = "," Then lngMoveToPosition = lngMoveToPosition + 1
                If ActiveDoc.Words(lngMoveToPosition) = "(" Then lngMoveToPosition = lngMoveToPosition + 1

                Call ErrorForm.LoadError("Dec DatePos", Citation.DateClause.DateStart, Citation.DateClause.DateEnd)
                If Not ErrorForm.IgnoreError Then
                    Call ErrorForm.Suggestion.Transpose(Citation.DateClause.DateStart, Citation.DateClause.DateEnd, lngMoveToPosition)
                    Call ErrorForm.Activate
                End If

            End If

        End If

    End If

End Sub

```

mCheckDate - 1

43

-- Copyright 2001 by Robert L. Jackson.

Option Explicit

Public Sub CheckDate()

```
Call CheckDateFormatting
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckMissingDate
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

End Sub

Private Sub CheckDateFormatting()

-- If the date clause specifies the month and date, but the date is formatted incorrectly, then suggest replacing the date clause with a date in the correct format.

!! TO DO: When support for unpublished reporters is added, check whether a full date is being used for a citation that requires only a year, or a year is being used for a citation that requires a full date.

```
Dim strDateCorrectFormat As String
Dim strDateAndYearOnly As String
Dim strMonthLong As String
Dim strMonthBluebook As String
```

-- Run this subroutine only if the date is a long date (not just a year), so it is longer than one word.

```
If Not (Citation.DateClause Is Nothing) Then
    If Citation.DateClause.DateEnd > Citation.DateClause.DateStart Then
```

-- Check whether the date is formatted correctly.

```
strDateCorrectFormat = DateBluebookFormat(Citation.DateClause.FullDate)
If Citation.DateClause.Text <> strDateCorrectFormat Then
```

```
    strDateAndYearOnly = Format$(Citation.DateClause.Text, " d, yyyy")
    strMonthBluebook = MonthBluebookFormat(Citation.DateClause.FullDate)
    strMonthLong = Format$(Citation.DateClause.FullDate, "mmmm")
```

-- Check whether the month is formatted incorrectly and the remainder of the date is formatted correctly. If the only problem is that the month is abbreviated incorrectly, then offer a specific error message suggesting only to replace the month and showing the table of correct abbreviations for months.

```
If Right$(Citation.DateClause.Text, Len(strDateAndYearOnly)) = strDateAndYearOnly _
And Not (strMonthBluebook = ActiveDoc.Words(Citation.DateClause.DateStart) _
Or strMonthBluebook = ActiveDoc.Words(Citation.DateClause.DateStart, Citation.DateClause.DateStart + 1)) Then
```

-- Display a different error message based on whether the Bluebook-formatted month is abbreviated or is not abbreviated.

```
If ActiveDoc.Words(Citation.DateClause.DateStart + 1) = "." Then
```

>> Jones v. Smith, 123 N.W.2d 456 (Minn. Jun. 10, 1980)

Call ErrorForm.LoadError("Dec DateFormatting A", Citation.DateClause.DateStart, Citation.DateClause.DateEnd, strMonthLong, strMonthBluebook)

If Not ErrorForm.IgnoreError Then

Call ErrorForm.Suggestion.Change(Citation.DateClause.DateStart, Citation.DateClause.DateEnd, strDateCorrectFormat)

Call ErrorForm.Activate

End If

Else

>> Jones v. Smith, 123 N.W.2d 456 (Minn. August 10, 1980)

Call ErrorForm.LoadError("Dec DateFormatting B", Citation.DateClause.DateStart, Citation.DateClause.DateEnd, strMonthLong, strMonthBluebook)

If Not ErrorForm.IgnoreError Then

Call ErrorForm.Suggestion.Change(Citation.DateClause.DateStart, Citation.DateClause.DateEnd, strDateCorrectFormat)

Call ErrorForm.Activate

End If

End If

-- Otherwise, raise a general error suggesting that the entire date clause be replaced with the correctly-formatted date.

>> Jones v. Smith, 123 N.W.2d 456 (Minn. 6/10/1980)

Else

Call ErrorForm.LoadError("Dec DateFormatting C", Citation.DateClause.DateStart, Citation.DateClause.DateEnd)

If Not ErrorForm.IgnoreError Then

Call ErrorForm.Suggestion.Change(Citation.DateClause.DateStart, Citation.DateClause.DateEnd, strDateCorrectFormat)

Call ErrorForm.Activate

End If

End If

End If

End If

End If

End Sub

Private Function DateBluebookFormat(SourceDate As Variant) As String

-- From the supplied date, returns a string with the date formatted in the correct Bluebook format -- e.g., "Aug. 3, 1970".

```
Dim strDateLongMonth As String
Dim strMonthLong As String
Dim strMonthBluebook As String
```

```
' -- First, format the string into "August 3, 1970" format using the Format function.
```

```
strDateLongMonth = Format$(SourceDate, "mmmm d, yyyy")
```

```
'-- Next, replace the long month with the correct abbreviation for the month. E.g., "August" with "Aug."
```

```
strMonthLong = Format$(SourceDate, "mmmm")
strMonthBluebook = MonthBluebookFormat(SourceDate)
If strMonthLong <> strMonthBluebook Then
    DateBluebookFormat = Replace(strDateLongMonth, strMonthLong, strMonthBluebook)
Else
    DateBluebookFormat = strDateLongMonth
End If
```

```
End Function
```

```
Private Function MonthBluebookFormat(SourceDate As Variant) As String
```

```
'-- Returns the correct Bluebook abbreviation for a given month. (See Bluebook page 298.)
```

```
Dim lngMonth As Long
```

```
lngMonth = CLng(Month(SourceDate))
MonthBluebookFormat = Choose(lngMonth, "Jan.", "Feb.", "Mar.", "Apr.", "May", "June", "July", "Aug.", "Sept.", "Oct.", "Nov.", "Dec.")
```

```
End Function
```

```
Private Sub CheckMissingDate()
```

```
'-- If there is no date clause, suggest inserting a placeholder for the year.
'>> Jones v. Smith, 123 N.W.2d 456 (Minn. Ct. App.)
```

```
' TO DO: When I add support for nonpublished cases, the program should suggest adding a placeholder for unpublished cases
' in the full date (mmmm dd, yyyy) format.
```

```
If Citation.DateClause Is Nothing Then
```

```
    Call ErrorForm.LoadError("Dec MissingDate", 0, 0)
    If Not ErrorForm.IgnoreError Then
        If Not (Citation.CourtClause Is Nothing) Then
            Call ErrorForm.Suggestion.Insert(Citation.CourtClause.CourtEnd + 1, "____")
        ElseIf Not (Citation.JurisdictionClause Is Nothing) Then
            Call ErrorForm.Suggestion.Insert(Citation.JurisdictionClause.JurEnd + 1, "____")
        Else
            Call ErrorForm.Suggestion.Insert(AppropriateDatePosition(Citation), "(____)")
        End If
        Call ErrorForm.Activate
    End If
```

```
End If
```

```
End Sub
```

```
'-- Copyright 2001 by Robert L. Jacobsen.
```

```
Option Explicit
```

```
Public Sub CheckEditor()
```

```
    Dim bcReporterClause As cClauseReporter
```

```
    For Each bcReporterClause In Citation.ReporterClauses
```

```
        Call CheckMissingEditorClause(bcReporterClause)
```

```
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
        Call CheckIncorrectEditorAbrv(bcReporterClause)
```

```
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
        Call CheckIncorrectEditor(bcReporterClause)
```

```
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
        Call CheckIncorrectEditorVolumeNumber(bcReporterClause)
```

```
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
        Call CheckMissingMainReporter(bcReporterClause)
```

```
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
        Call CheckParallelEditorShouldBeParenthetical(bcReporterClause)
```

```
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
    Next bcReporterClause
```

```
End Sub
```

```
Private Sub CheckMissingEditorClause(CurrentClause As cClauseReporter)
```

```
    '-- If the reporter clause should have an editor parenthetical, but doesn't, then suggest inserting an editor parenthetical.
```

```
    '>> Jones v. Smith, 40 U.S. 500, 505 (1839)
```

```
    '>> Jones v. Smith, 2 Ky. 500, 505 (1803)
```

```
    Dim bcExpectedReporter As cReporter
```

```
    Dim lngExpectedVolumeNumber As Long
```

```
    If CurrentClause.EditorClause Is Nothing Then
```

```
        Set bcExpectedReporter = ExpectedEditorReporter(CurrentClause)
```

```
        If Not (bcExpectedReporter Is Nothing) Then
```

```
            lngExpectedVolumeNumber = ExpectedVolumeNumber(CurrentClause, bcExpectedReporter)
```

```
            Call ErrorForm.LoadError("Rep MissingEditorParenthetical", CurrentClause.ClauseStart, CurrentClause.ClauseEnd, ActiveDoc.Words(
```

```
CurrentClause.VolumeNumberLocation), CurrentClause.Reporter.FullName, bcExpectedReporter.FullName)
```

```
            If Not ErrorForm.IgnoreError Then
```

```
                Call ErrorForm.Suggestion.Insert(CurrentClause.AbrvEnd + 1, ParentheticalText(lngExpectedVolumeNumber, bcExpectedReporter))
```

```
                Call ErrorForm.Activate
```

```
            End If
```

```
        End If
```

```
    End If
```

```
End Sub
```

```
Private Sub CheckIncorrectEditorAbrv(CurrentClause As cClauseReporter)
```

```
    '-- If the editor reporter is not punctuated or spelled correctly, then suggest correcting it.
```

```
    '>> Jones v. Smith, 69 U.S. (2 Wall) 500, 505 (1870)
```

```
    If Not (CurrentClause.EditorClause Is Nothing) Then
```

```
        With CurrentClause.EditorClause
```

```
            If .AbrvText <> .Reporter.AbrvName Then
```

```
                Call ErrorForm.LoadError("Rep IncorrectEditorAbrv", .AbrvStart, .AbrvEnd, .Reporter.FullName)
```

```
                If Not ErrorForm.IgnoreError Then
```

```
                    Call ErrorForm.Suggestion.Change(.AbrvStart, .AbrvEnd, .Reporter.AbrvName)
```

```
                    Call ErrorForm.Activate
```

```
                End If
```

```
            End If
```

```
        End With
```

```
    End If
```

```
End Sub
```

```
Private Sub CheckIncorrectEditor(CurrentClause As cClauseReporter)
```

```
    '-- If the current clause has an associated editor clause, but the editor clause is not a correct editor for the  
    ' current clause, then suggest deleting the editor clause if an editor clause is not required, or changing the editor  
    ' to the correct editor if one is required.
```

```
    '>> Jones v. Smith, 200 U.S. (10 S. Ct) 500, 505 (1980)
```

```
    '>> Jones v. Smith, 50 U.S. (10 S. Ct.) 500, 505 (1805)
```

```
    Dim bcExpectedReporter As cReporter
```

```
    Dim lngExpectedVolumeNumber As Long
```

```
    If Not (CurrentClause.EditorClause Is Nothing) And CurrentClause.VolumeNumber > 0 Then  
        With CurrentClause.EditorClause
```

```
Set bcExpectedReporter = ExpectedEditorReporter(CurrentClause)
```

```
'-- If the reporter should not have an editor clause, then suggest deleting it.
If bcExpectedReporter Is Nothing Then
```

```
Call ErrorForm.LoadError("Rep IncorrectEditor A", .ClauseStart, .ClauseEnd, .Reporter.FullName, CurrentClause.Reporter.Full
```

```
Name)
If Not ErrorForm.IgnoreError Then
Call ErrorForm.Suggestion.Delete(.ClauseStart, .ClauseEnd)
Call ErrorForm.Activate
End If
```

```
'-- If reporter should have an editor clause, but the current editor reporter is incorrect, then suggest changing it.
ElseIf Not (bcExpectedReporter Is .Reporter) Then
lngExpectedVolumeNumber = ExpectedVolumeNumber(CurrentClause, bcExpectedReporter)
```

```
Call ErrorForm.LoadError("Rep IncorrectEditor B", .ClauseStart, .ClauseEnd, .Reporter.FullName, CurrentClause.Reporter.Full
```

```
Name)
If Not ErrorForm.IgnoreError Then
Call ErrorForm.Suggestion.Change(.ClauseStart, .ClauseEnd, ParentheticalText(lngExpectedVolumeNumber, bcExpectedReporte
r))
Call ErrorForm.Activate
End If
```

```
End If
```

```
End With
```

```
End If
```

```
End Sub
```

```
Private Sub CheckIncorrectEditorVolumeNumber(CurrentClause As cClauseReporter)
```

```
'-- If the current clause has an editor clause and the volume number does not correspond with the volume number for the
' parallel main reporter clause, then suggest correcting the editor clause's volume number.
```

```
Dim lngExpectedVolumeNumber As Long
Dim strSuggestedChange As String
```

```
If Not (CurrentClause.EditorClause Is Nothing) And CurrentClause.VolumeNumber > 0 Then
```

```
With CurrentClause.EditorClause
```

```
'-- Does the editor clause have the correct editor for this main clause's volume number and reporter?
If ExpectedEditorReporter(CurrentClause) Is .Reporter Then
```

```
lngExpectedVolumeNumber = ExpectedVolumeNumber(CurrentClause, .Reporter)
```

```
'-- Is the editor clause missing a volume number when one is required?
```

```


```

```
If .VolumeNumberLocation = 0 And lngExpectedVolumeNumber > 0 Then
```

```
Call ErrorForm.LoadError("Rep IncorrectEditorVolumeNumber 1", .AbrvStart, .AbrvEnd, .Reporter.FullName)
If Not ErrorForm.IgnoreError Then
Call ErrorForm.Suggestion.Insert(.AbrvStart, lngExpectedVolumeNumber)
Call ErrorForm.Activate
End If
```

```
'-- Does the editor clause have a volume number when none is required?
```

```


```

```
ElseIf .VolumeNumberLocation > 0 And lngExpectedVolumeNumber = 0 Then
```

```
Call ErrorForm.LoadError("Rep IncorrectEditorVolumeNumber 2", .VolumeNumberLocation, .VolumeNumberLocation, .Reporter.F
ullName, CurrentClause.Reporter.FullName)
If Not ErrorForm.IgnoreError Then
Call ErrorForm.Suggestion.Delete(.VolumeNumberLocation, .VolumeNumberLocation)
Call ErrorForm.Activate
End If
```

```
'-- Is the editor clause's volume number inconsistent with the correct volume number?
```

```


```

```
ElseIf .VolumeNumber > 0 And .VolumeNumber <> lngExpectedVolumeNumber Then
```

```
Call ErrorForm.LoadError("Rep IncorrectEditorVolumeNumber 3", .VolumeNumberLocation, .AbrvEnd, CStr(lngExpectedVolumeNu
mber), .Reporter.FullName, ActiveDoc.Words(CurrentClause.VolumeNumberLocation, CurrentClause.AbrvEnd))
If Not ErrorForm.IgnoreError Then
strSuggestedChange = lngExpectedVolumeNumber & " " & ActiveDoc.Words(.AbrvStart, .AbrvEnd)
Call ErrorForm.Suggestion.Change(.VolumeNumberLocation, .AbrvEnd, strSuggestedChange)
Call ErrorForm.Activate
End If
```

```
End If
```

```
End If
```

```
End With
```

```
End If
```

```
End Sub
```

```
Private Sub CheckMissingMainReporter(CurrentClause As cClauseReporter)
```

```
'-- If the citation contains a editor reporter clause but is missing an associated main reporter clause, then suggest
' inserting a main reporter clause.
```

```


```

```
Dim bcMainReporter As cReporter
Dim lngMainVolumeNumber As Long
Dim bcEditor As cReporter
```

```

If CurrentClause.Reporter.IsEditorReporter Then
    With CurrentClause
        If MainReporterClause(CurrentClause) Is Nothing Then
            Set bcEditor = .Reporter
            Set bcMainReporter = ExpectedMainReporter(CurrentClause)
            If Not (bcMainReporter Is Nothing) Then
                lngMainVolumeNumber = .VolumeNumber + (bcEditor.MainReporterVolumeFirst - bcEditor.VolumeStart)

                Call ErrorForm.LoadError("Rep MissingMainReporter", CurrentClause.ClauseStart, CurrentClause.ClauseEnd, bcEditor.FullName, bcMainReporter.FullName)
                If Not ErrorForm.IgnoreError Then
                    '-- Suggest adding a main reporter clause with the current editor as a parenthetical.
                    Call ErrorForm.Suggestion.Insert(.ClauseStart, CStr(lngMainVolumeNumber) & " " & bcMainReporter.AbrvName & " (")
                    Call ErrorForm.Suggestion.Insert(.AbrvEnd + 1, ")")
                    Call ErrorForm.DisplaySuggestion

                    '-- Suggest adding the main reporter clause as a parallel citation.
                    Call ErrorForm.Suggestion.Insert(CurrentClause.ClauseStart, CStr(lngMainVolumeNumber) & " " & bcMainReporter.AbrvName & "_,")

                    Call ErrorForm.Activate
                End If
            End If
        End If
    End With
End If

End Sub

Private Sub CheckParallelEditorShouldBeParenthetical(ReporterClause As cClauseReporter)
    '-- If the citation has both a main reporter clause and an editor clause as a separate parallel citation, but the editor
    '-- clause's pagination is the same as the main reporter clause, then suggest changing the parallel citation to a
    '-- parenthetical citation.
    '-- Jones v. Smith, 69 U.S. 500, 505, 2 Wall. 500 (1870)

    Dim bcMainReporterClause As cClauseReporter
    Dim strParentheticalText As String

    If ReporterClause.Reporter.IsEditorReporter Then
        With ReporterClause
            Set bcMainReporterClause = MainReporterClause(ReporterClause)
            If Not (bcMainReporterClause Is Nothing) Then
                If .FirstPageNumber > 0 _
                    And .FirstPageNumber = bcMainReporterClause.FirstPageNumber Then
                    strParentheticalText = ParentheticalText(.VolumeNumber, .Reporter)

                    Call ErrorForm.LoadError("Rep ParallelEditorShouldBeParenthetical", .ClauseStart, .ClauseEnd, bcMainReporterClause.Reporter.FullName, .Reporter.FullName)
                    If Not ErrorForm.IgnoreError Then
                        Call ErrorForm.Suggestion.Delete(.ClauseStart, .ClauseEnd)
                        Call ErrorForm.Suggestion.Insert(bcMainReporterClause.AbrvEnd + 1, strParentheticalText)
                        Call ErrorForm.Activate
                    End If
                End If
            End If
        End With
    End If

End Sub

Private Function ExpectedEditorReporter(MainReporterClause As cClauseReporter) As cReporter
    '-- Identify the correct editor reporter for this main reporter.

    Dim bcEditorReporter As cReporter

    If MainReporterClause.VolumeNumber > 0 Then
        For Each bcEditorReporter In MainReporterClause.Reporter.Editors
            '-- Does this editor have the correct range of volume numbers for the main reporter's volume number?
            If MainReporterClause.VolumeNumber >= bcEditorReporter.MainReporterVolumeFirst And MainReporterClause.VolumeNumber <= bcEditorReporter.MainReporterVolumeLast Then
                Set ExpectedEditorReporter = bcEditorReporter
                Exit Function
            End If
        Next bcEditorReporter
    End If

End Function

Private Function ExpectedVolumeNumber(ReporterClause As cClauseReporter, EditorReporter As cReporter) As Long
    '-- Return the volume number for an editor that is expected based on the volume number of the main reporter.

    If ReporterClause.VolumeNumber > 0 Then
        If EditorReporter.VolumeStart = EditorReporter.VolumeEnd Then
            ExpectedVolumeNumber = 0
        End If
    End If
End Function

```



```

Else
    ExpectedVolumeNumber = ReporterClause.VolumeNumber - EditorReporter.MainReporter.VolumeFirst + EditorReporter.VolumeStart
End If
End If

End Function

Private Function ExpectedMainReporter(EditorReporterClause As cClauseReporter) As cReporter

    '-- Identify the main reporter that is associated with the supplied editor clause.

    Dim bcReporter As cReporter
    Dim bcEditorReporter As cReporter
    Dim bcCurrentEditor As cReporter

    Set bcCurrentEditor = EditorReporterClause.Reporter
    For Each bcReporter In EditorReporterClause.Reporter.Parent.Reporters
        '-- Does this reporter have any associated editor reporters that matches the current editor reporter?
        For Each bcEditorReporter In bcReporter.Editors
            If bcEditorReporter Is bcCurrentEditor Then
                Set ExpectedMainReporter = bcReporter
                Exit Function
            End If
        Next bcEditorReporter
    Next bcReporter

End Function

Private Function MainReporterClause(EditorReporterClause As cClauseReporter) As cClauseReporter

    '-- Returns the main reporter clause that is associated with this editor reporter clause.

    Dim bcReporterClause As cClauseReporter
    Dim bcEditor As cReporter

    For Each bcReporterClause In Citation.ReporterClauses
        For Each bcEditor In bcReporterClause.Reporter.Editors
            If bcEditor Is EditorReporterClause.Reporter Then
                Set MainReporterClause = bcReporterClause
                Exit Function
            End If
        Next bcEditor
    Next bcReporterClause

End Function

Private Function ParentheticalText(ByVal VolumeNumber As Long, ByVal Editor As cReporter) As String

    Returns the correct volume/editor name combination that should be used in a parenthetical, based on the main
    reporter clause and the editor to be inserted.

    If VolumeNumber = 0 Then
        ParentheticalText = "(" & Editor.AbrvName & ")"
    Else
        ParentheticalText = "(" & VolumeNumber & " " & Editor.AbrvName & ")"
    End If

End Function

```

mCheckJurAndCourt - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Public Sub CheckJurAndCourt()

```

    Call CheckMissingJurisdiction
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

```

```

    Call CheckMissingCourt
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

```

```

    Call CheckInconsistentJurisdictionAndCourt
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

```

```

    Call CheckExtraneousJurisdiction
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

```

```

    Call CheckExtraneousCourtAndJurisdiction
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

```

```

    Call CheckJurisdictionAndCourtSpelling
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

```

```

    Call CheckValidYear
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

```

End Sub

Private Sub CheckMissingJurisdiction()

```

'-- If the jurisdiction could not be determined, then suggest adding a jurisdiction clause for every jurisdiction that is a
'   good match for the reporters, court clause and date. Also suggest adding a court clause if the court could not be
'   determined.

```

```

Dim colCompatibleJurisdictions As Collection
Dim bcJurisdiction As cJurisdiction
Dim colCompatibleCourts As Collection
Dim bcCourt As cCourt

```

```

If Citation.Jurisdiction Is Nothing And Citation.JurisdictionClause Is Nothing Then

```

```

    If Not (Citation.CourtClause Is Nothing) Then

```

```

        '-- If there is a recognized court clause in the citation, then only suggest adding a jurisdiction.
        '>> Jones v. Smith, 123 N.W.2d 456 (Ct. App. 1980)
        '>> Jones v. Smith, 123 Misc.2d 456 (Ct. Cl. 1980)

```

```

        Set colCompatibleJurisdictions = CompatibleJurisdictions
        Select Case colCompatibleJurisdictions.Count
            Case 0
                Call ErrorForm.LoadError("Dec MissingJurisdiction B1", Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd)
            Case 1
                Call ErrorForm.LoadError("Dec MissingJurisdiction B2", Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd)
            Case Is > 1
                Call ErrorForm.LoadError("Dec MissingJurisdiction B3", Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd)
        End Select

```

```

        If Not ErrorForm.IgnoreError Then
            For Each bcJurisdiction In colCompatibleJurisdictions
                Call ErrorForm.Suggestion.Change(Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd, bcJurisdiction.AbrvNam
e & Citation.CourtClause.Text)
                Call ErrorForm.DisplaySuggestion
            Next bcJurisdiction
            Call ErrorForm.Activate
        End If

```

```

    Else

```

```

        '-- If there is no recognized court clause, then suggest adding a jurisdiction clause and court clause.
        '>> Jones v. Smith, 123 N.W.2d 456 (1980)
        '>> Jones v. Smith, 123 A.D.2d 456 (1980)

```

```

        Set colCompatibleCourts = CompatibleCourts
        Select Case colCompatibleCourts.Count
            Case 0
                Call ErrorForm.LoadError("Dec MissingJurisdiction A1", 0, 0)
            Case 1
                Call ErrorForm.LoadError("Dec MissingJurisdiction A2", 0, 0)
            Case Is > 1
                Call ErrorForm.LoadError("Dec MissingJurisdiction A3", 0, 0)
        End Select

```

```

        If Not ErrorForm.IgnoreError Then
            For Each bcCourt In colCompatibleCourts
                Call ErrorForm.Suggestion.Insert(AppropriateJurisdictionPosition(Citation), JurisdictionAndCourtText(bcCourt))
                Call ErrorForm.DisplaySuggestion
            Next bcCourt
            Call ErrorForm.Activate
        End If

```

```

    End If

```

```

End If

```

End Sub

Private Function JurisdictionAndCourtText(Court As cCourt)

```

    If Court.JurisdictionImplicit _

```

```

Or Not (Citation.CourtClause Is Nothing) Then
    JurisdictionAndCourtText = Citation.CourtClause.Court.Parent.AbrvName
Else
    JurisdictionAndCourtText = Citation.CourtClause.Court.Parent.AbrvName & " " & Court.AbrvName
End If

```

```
End Function
```

```
Private Sub CheckMissingCourt()
```

```

'-- If the jurisdiction was determined though parse but the court was not and there is no court clause, then suggest adding a
' court clause for every court that is a good match for the reporters and date.
'>> Jones v. Smith, 123 F.2d 456 (1980)

```

```

Dim colCompatibleCourts As Collection
Dim bcCourt As cCourt
Dim lngInsertionPosition As Long

```

```

If Not Citation.Jurisdiction Is Nothing Then
    If Citation.Court Is Nothing And Citation.CourtClause Is Nothing Then

```

```

        Set colCompatibleCourts = CompatibleCourts(Citation.Jurisdiction)
        Select Case colCompatibleCourts.Count
            Case 0
                Call ErrorForm.LoadError("Dec MissingCourt 1", 0, 0)
            Case 1
                Call ErrorForm.LoadError("Dec MissingCourt 2", 0, 0)
            Case Is > 1
                Call ErrorForm.LoadError("Dec MissingCourt 3", 0, 0)
        End Select

```

```

        If Not ErrorForm.IgnoreError Then
            lngInsertionPosition = AppropriateCourtPosition(Citation)
            For Each bcCourt In colCompatibleCourts
                Call ErrorForm.Suggestion.Insert(lngInsertionPosition, bcCourt.AbrvName)
                Call ErrorForm.DisplaySuggestion
            Next bcCourt
            Call ErrorForm.Activate
        End If
    End If
End Sub

```

```
Private Sub CheckInconsistentJurisdictionAndCourt()
```

```

'-- If the court clause is inconsistent with the citation's jurisdiction, then suggest either replacing the court clause
' with a compatible jurisdiction, or replacing the jurisdiction clause with a compatible court.

```

```

NB: This sub ONLY checks when there is a court clause that disagrees with a jurisdiction clause.

```

```

Dim bcCourt As cCourt
Dim bcJurisdiction As cJurisdiction

```

```

Is the court clause's parent jurisdiction different than the jurisdiction clause's jurisdiction?

```

```

If Not (Citation.JurisdictionClause Is Nothing) And Not (Citation.CourtClause Is Nothing) Then
    If Not (Citation.CourtClause.Court.Parent Is Citation.JurisdictionClause.Jurisdiction) Then

```

```

        '-- If the court clause's parent jurisdiction is federal and a state jurisdiction clause immediately follows the court clause,
        ' do not flag an error -- this is handled in CheckExtraneousJurisdictionOrCourt.

```

```

        If Not (Citation.CourtClause.Court.Parent.FullName = "Federal" _
            And Citation.JurisdictionClause.JurStart = ActiveDoc.Words.NextFullWord(Citation.CourtClause.CourtEnd)) Then

```

```

            If Citation.CourtClause.Court.Parent Is Citation.Jurisdiction Then
                '>> Jones v. Smith, 123 F.2d 456 (Minn. 2d Cir. 1990)

```

```

                Call ErrorForm.LoadError("Dec InconsistentJurAndCourt 1", Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd, Citation.Jurisdiction.FullName, Citation.CourtClause.Court.FullName, Citation.CourtClause.Text, Citation.JurisdictionClause.Jurisdiction.FullName)

```

```

            ElseIf Citation.CourtClause.Court.JurisdictionImplicit Then

```

```

                '>> Jones v. Smith, 123 N.W.2d 456 (Minn. 2d Cir. 1990)
                Call ErrorForm.LoadError("Dec InconsistentJurAndCourt 2", Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd, Citation.Jurisdiction.FullName, Citation.JurisdictionClause.Text, Citation.CourtClause.Court.FullName)

```

```

            Else
                '>> Jones v. Smith, 123 N.W.2d 456 (Minn. Ct. Cl. 1990)

```

```

                Call ErrorForm.LoadError("Dec InconsistentJurAndCourt 3", Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd, Citation.Jurisdiction.FullName, Citation.JurisdictionClause.Text)
            End If

```

```

        If Not ErrorForm.IgnoreError Then

```

```

            '-- Suggest replacing or deleting the jurisdiction.

```

```

            If Citation.CourtClause.Court.JurisdictionImplicit Then

```

```

                Call ErrorForm.Suggestion.Delete(Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd)
                Call ErrorForm.DisplaySuggestion
            Else

```

```

                For Each bcJurisdiction In CompatibleJurisdictions

```

```

                    Call ErrorForm.Suggestion.Change(Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd, bcJurisdiction.AbrvName)

```

```

                Call ErrorForm.DisplaySuggestion

```

```

                Next bcJurisdiction

```

```

            End If

```

```

        '-- Suggest replacing the court clause.

```

```

        For Each bcCourt In CompatibleCourts(Citation.JurisdictionClause.Jurisdiction)

```

```

            If bcCourt.AbrvName = "" Then

```

```

                Call ErrorForm.Suggestion.Delete(Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd)
            End If
        End For
    End If
End Sub

```

```

        Call ErrorForm.DisplaySuggestion
    Else
        Call ErrorForm.Suggestion.Change(Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd, bcCourt.AbrvName)
    End If
    Call ErrorForm.DisplaySuggestion
End If
Next bcCourt

Call ErrorForm.Activate

End If

End If

End If

End Sub

Private Sub CheckExtraneousJurisdiction()

    '-- Suggest deleting the jurisdiction clause if the jurisdiction is redundant because it is identified by the court clause.

    If Not (Citation.JurisdictionClause Is Nothing) And Not (Citation.CourtClause Is Nothing) Then

        If CourtClauseIDsJurisdiction() Then

            If Citation.Jurisdiction.FullName = "Federal" _
            And Citation.JurisdictionClause.Jurisdiction.FullName = "Federal" Then

                '-- If a federal jurisdiction clause is redundant because it is identified by a federal court clause, then suggest deleting
                ' the jurisdiction clause.
                '>> Jones v. Smith, 123 F.2d 456 (Fed. 8th Cir. 1980)

                Call ErrorForm.LoadError("Dec ExtraneousJurisdiction A", Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.
JurEnd, Citation.Court.FullName)
                If Not ErrorForm.IgnoreError Then
                    Call ErrorForm.Suggestion.Delete(Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd)
                    Call ErrorForm.Activate
                End If

                ElseIf Citation.Jurisdiction.FullName = "Federal" _
                And Citation.JurisdictionClause.Jurisdiction.FullName <> "Federal" _
                And ClausesAreAdjacent(Citation.JurisdictionClause, Citation.CourtClause) Then

                    '-- If a non-federal jurisdiction clause is inconsistent with a federal court clause, and the jurisdiction clause
                    ' immediately follows a federal court clause, then suggest deleting the jurisdiction clause. (If the jurisdiction clause
                    ' does not immediately follow the court clause, that error is handled by CheckInconsistentJurisdictionAndCourt.)
                    '>> Jones v. Smith, 123 F.2d 456 (8th Cir. (Minn.) 1980)

                    Call ErrorForm.LoadError("Dec ExtraneousJurisdiction B", Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.
JurEnd, Citation.Court.FullName)
                    If Not ErrorForm.IgnoreError Then
                        Call ErrorForm.Suggestion.Delete(Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd)
                        Call ErrorForm.Activate
                    End If

                    ElseIf Citation.Jurisdiction.FullName <> "Federal" _
                    And Citation.JurisdictionClause.Jurisdiction Is Citation.Jurisdiction Then

                        '-- If the state jurisdiction clause is redundant because it is identified by the state court clause, then suggest deleting
                        ' the jurisdiction clause.
                        '>> Jones v. Smith, 123 N.W. 456 (N.D. Dakota 1880)

                        Call ErrorForm.LoadError("Dec ExtraneousJurisdiction C", Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.
JurEnd, Citation.Court.FullName)
                        If Not ErrorForm.IgnoreError Then
                            Call ErrorForm.Suggestion.Delete(Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd)
                            Call ErrorForm.Activate
                        End If

                    End If

                End If

            End If

        End Sub

    Private Sub CheckExtraneousCourtAndJurisdiction()

        '-- Suggest deleting the court and/or jurisdiction if either is identified by a reporter clause.

        Dim bcReporterIDingJur As cReporter
        Dim bcReporterIDingCourt As cReporter
        Dim bolExtraneousJurisdiction As Boolean
        Dim bolExtraneousCourt As Boolean
        Dim bolFixCourtAndJurTogether As Boolean

        If Not (Citation.JurisdictionClause Is Nothing) Then
            Set bcReporterIDingJur = ReporterIDingJurisdiction(Citation.Jurisdiction)
            If Not (bcReporterIDingJur Is Nothing) Then bolExtraneousJurisdiction = True
        End If

        If Not (Citation.CourtClause Is Nothing) Then
            Set bcReporterIDingCourt = ReporterIDingCourt(Citation.Court)
            If Not (bcReporterIDingCourt Is Nothing) Then bolExtraneousCourt = True
        End If

        If bolExtraneousJurisdiction And bolExtraneousCourt Then

```

```

    If Citation.JurisdictionClause.NextClause Is Citation.CourtClause Then bolFixCourtAndJurTogether = True
End If

If bolFixCourtAndJurTogether = True Then
    '-- If both the jurisdiction and court are redundant because they are identified by a reporter clause, and the court
    ' clause immediately follows the jurisdiction clause, then suggest deleting the jurisdiction clause and court clause
    ' together.
    '>> Jones v. Smith, 123 Ill. App. 2d 456 (Ill. App. Ct. 1980)
    '>> Jones v. Smith, 123 Ct. Cl. 456 (Fed. Ct. Cl. 1980)

    Call ErrorForm.LoadError("Dec ExtraneousCourtAndJurisdiction A", Citation.JurisdictionClause.JurStart, Citation.CourtClause.CourtEnd, Citation.Court.FullName, bcReporterIDingJur.FullName)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Delete(Citation.JurisdictionClause.JurStart, Citation.CourtClause.CourtEnd)
        Call ErrorForm.Activate
    End If

Else

    If bolExtraneousJurisdiction Then

        '-- If the jurisdiction is redundant because it is identified by a reporter clause, then suggest deleting the
        ' jurisdiction clause.
        '>> Jones v. Smith, 123 U.S. 456 (Fed. 1980)
        '>> Jones v. Smith, 123 Ill. App. 2d 456 (Ill. 1980)
        '>> Jones v. Smith, 123 Dakota 456 (N.D. 1880)

        Call ErrorForm.LoadError("Dec ExtraneousCourtAndJurisdiction B", Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd, bcReporterIDingJur.FullName)
        If Not ErrorForm.IgnoreError Then
            Call ErrorForm.Suggestion.Delete(Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd)
            Call ErrorForm.Activate
            If ErrorForm.Result = Cancel Or ErrorForm.Result = Change Then Exit Sub
        End If

    ElseIf bolExtraneousCourt Then

        '-- If the court clause is redundant because the court is designated by a reporter clause, then suggest deleting the
        ' court clause.
        '>> Jones v. Smith, 123 Ill. App. 2d 456 (App. Ct. 1980)
        '>> Jones v. Smith, 123 Ct. Cl. 456 (Ct. Cl. 1980)

        Call ErrorForm.LoadError("Dec ExtraneousCourtAndJurisdiction A", Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd, Citation.Court.FullName, bcReporterIDingCourt.FullName)
        If Not ErrorForm.IgnoreError Then
            Call ErrorForm.Suggestion.Delete(Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd)
            Call ErrorForm.Activate
        End If
    End If
End If

End Sub

Private Sub CheckJurisdictionAndCourtSpelling()
    Dim strCorrectJurAbv As String
    Dim strCorrectCourtAbv As String
    Dim bolFixJurisdiction As Boolean
    Dim bolFixCourt As Boolean
    Dim bolFixJurAndCourtTogether As Boolean

    '-- Determine whether the jurisdiction clause should be corrected.
    If Not (Citation.JurisdictionClause Is Nothing) Then
        strCorrectJurAbv = Citation.JurisdictionClause.Jurisdiction.AbrvName
        If Citation.JurisdictionClause.Text <> strCorrectJurAbv Then bolFixJurisdiction = True
    End If

    '-- Determine whether the court clause should be corrected.
    If Not (Citation.CourtClause Is Nothing) Then
        strCorrectCourtAbv = Citation.CourtClause.Court.AbrvName
        If Citation.CourtClause.Text <> strCorrectCourtAbv Then bolFixCourt = True
    End If

    '-- If either needs to be corrected, then determine whether the jurisdiction clause is adjacent to the court clause and the
    ' court clause is compatible with the jurisdiction.
    If bolFixJurisdiction Or bolFixCourt Then
        If Not (Citation.CourtClause Is Nothing) And Not (Citation.JurisdictionClause Is Nothing) Then
            If Citation.CourtClause.Court.Parent Is Citation.JurisdictionClause.Jurisdiction _
            And Citation.JurisdictionClause.NextClause Is Citation.CourtClause Then
                bolFixJurAndCourtTogether = True
            End If
        End If
    End If

    If bolFixJurAndCourtTogether = True Then

        '-- If the court Clause is adjacent to the jurisdiction clause and either or both clauses needs to be corrected, then
        ' suggest correcting the two clauses simultaneously (as a single unit.)
        '>> Jones v. Smith, 123 N.W.2d 456 (Minn. Ct App. 1980)

        Call ErrorForm.LoadError("Dec JurisdictionAndCourtIDSpelling A", Citation.JurisdictionClause.JurStart, Citation.CourtClause.CourtEnd, Citation.CourtClause.Court.FullName)
        If Not ErrorForm.IgnoreError Then
            Call ErrorForm.Suggestion.Change(Citation.JurisdictionClause.JurStart, Citation.CourtClause.CourtEnd, strCorrectJurAbv & " " & strCorrectCourtAbv)
            Call ErrorForm.Activate
        End If
    End If

Else

```

```

'-- Otherwise, suggest separate corrections for the jurisdiction and the court if appropriate.
'-- Suggest correcting the jurisdiction if appropriate.
'>> Jones v. Smith, 123 N.W.2d 456 (Minn. 1980)

If bolFixJurisdiction = True Then
    Call ErrorForm.LoadError("Dec JurisdictionAndCourtIDSpelling B", Citation.JurisdictionClause.JurStart, Citation.JurisdictionCla
use.JurEnd, Citation.JurisdictionClause.Jurisdiction.FullName)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Change(Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd, strCorrectJurAbr
v)
        Call ErrorForm.Activate
        If ErrorForm.Result = Cancel Or ErrorForm.Result = Change Then Exit Sub
    End If
End If

'-- Suggest correcting the court if appropriate.
'>> Jones v. Smith, 123 F.2d 456 (C.C.A.2d 1980)

If bolFixCourt = True Then
    Call ErrorForm.LoadError("Dec JurisdictionAndCourtIDSpelling C", Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd
, Citation.CourtClause.Court.FullName)
    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Change(Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd, strCorrectCourtAbvr)
        Call ErrorForm.Activate
        If ErrorForm.Result = Cancel Or ErrorForm.Result = Change Then Exit Sub
    End If
End If

End If

End Sub

Private Sub CheckValidYear()
    'If the designated year is not within a valid range for the designated court, then request the user to correct the year.
    'Jones v. Smith, 123 N.W.2d 456 (Minn. Ct. App. 1950)
    If Not (Citation.Court Is Nothing) Then
        If Citation.Year > 0 And Not (Citation.Court.YearStart <= Citation.Year And Citation.Year <= Citation.Court.YearEnd) Then
            Call ErrorForm.LoadError("Dec ValidYear A", Citation.DateClause.DateEnd, Citation.DateClause.DateEnd, Citation.DateClause.Year,
Citation.Court)
            If Not ErrorForm.IgnoreError Then
                Call ErrorForm.Suggestion.Change(Citation.DateClause.DateStart, Citation.DateClause.DateEnd, "____")
                Call ErrorForm.Activate
            End If
        End If
    End If
End Sub

Private Function CourtClauseIDsJurisdiction() As Boolean
    If Not (Citation.CourtClause Is Nothing) Then
        If Citation.CourtClause.Court.JurisdictionImplicit = True Then
            If Citation.CourtClause.Court.Parent Is Citation.Jurisdiction Then
                CourtClauseIDsJurisdiction = True
            End If
        End If
    End If
End Function

Private Function ReporterIDingJurisdiction(Jurisdiction As cJurisdiction) As cReporter
    Dim bcReporterClause As cClauseReporter

    For Each bcReporterClause In Citation.ReporterClauses
        If bcReporterClause.Reporter.ImplicitJurisdiction Is Jurisdiction Then
            Set ReporterIDingJurisdiction = bcReporterClause.Reporter
            Exit Function
        End If
    Next bcReporterClause

End Function

Private Function ReporterIDingCourt(Court As cCourt) As cReporter
    Dim bcReporterClause As cClauseReporter

    For Each bcReporterClause In Citation.ReporterClauses
        If bcReporterClause.Reporter.ImplicitCourt Is Court Then
            Set ReporterIDingCourt = bcReporterClause.Reporter
            Exit Function
        End If
    Next bcReporterClause

End Function

Private Function CompatibleJurisdictions() As Collection

```

```

Dim strCourtClauseTextTrim As String
Dim bcJurisdiction As cJurisdiction
Dim bcMatchingCourt As cCourt
Dim colJurisdictions As Collection
Dim i As Integer

'-- Initialize the data.

Set colJurisdictions = New Collection
strCourtClauseTextTrim = Citation.CourtClause.Text
Jurisdictions.ScoreReset

'-- Step through each jurisdiction.
For Each bcJurisdiction In Jurisdictions

    '-- If there is a court in this jurisdiction with an alias that matches the court clause, then calculate points for the
    ' jurisdiction.
    Set bcMatchingCourt = bcJurisdiction.Courts.Item(strCourtClauseTextTrim, True, Citation.DateClause.Year, True)
    If Not (bcMatchingCourt Is Nothing) Then
        '-- Award one point to show that the jurisdiction has a matching court alias.
        bcJurisdiction.Score = bcJurisdiction.Score + 1
        '-- Award two points for every reporter clause that is compatible with the jurisdiction.
        bcJurisdiction.Score = bcJurisdiction.Score + 2 * ReportersCompatibleWithCourt(bcMatchingCourt, Citation.ReporterClauses)
        '-- If there are no reporter clauses, then award one point if the citation's year is within the court's date range.
        ' (If there are reporter clauses, then do not perform this test because points were awarded for a similar test in
        ' CompatibilityWithReporters.)
        If Citation.ReporterClauses.Count = 0 Then If Citation.Year > 0 And bcMatchingCourt.YearStart <= Citation.Year And Citation.Yea
r <= bcMatchingCourt.YearEnd Then bcJurisdiction.Score = bcJurisdiction.Score + 1
        '-- Award one point if the court clause exactly matches the correct abbreviation for the matching court (i.e., is not
        ' an incorrect alias.)
        If bcMatchingCourt.AbrvNameTrim = strCourtClauseTextTrim Then bcJurisdiction.Score = bcJurisdiction.Score + 1
        '-- Determine whether this jurisdiction's score is the best score so far.
    End If

Next bcJurisdiction

'-- Return a collection of the jurisdictions that have a score within one point of the best score. (A permitted deviation of
' one points includes a jurisdiction if the court name was wrong or the year was wrong, compared to the best matches, but not
' if both were wrong.)
Set CompatibleJurisdictions = Jurisdictions.ScoresBest(2, 1)

End Function

Private Function CompatibleCourts(Optional Jurisdiction As cJurisdiction) As Collection

    ' The jurisdiction is the declaration is optional. If jurisdiction is supplied, then only search for compatible courts
    ' within that jurisdiction. Otherwise, search for compatible courts within all jurisdictions.

    Dim bcCourtsToCheck As cCourts
    Dim lngMinimumScore As Long
    Dim bcCourt As cCourt
    Dim bcUniqueCourt As cCourt
    Dim colCourts As Collection
    Dim colUniqueCourts As Collection
    Dim bolIsUnique As Boolean

    ' Determine the correct group of courts to search.
    If Jurisdiction Is Nothing Then
        Set bcCourtsToCheck = Jurisdictions.AllCourts
        lngMinimumScore = 2
    Else
        Set bcCourtsToCheck = Jurisdiction.Courts
        lngMinimumScore = 1
    End If

    '-- Step through each court and award points.
    bcCourtsToCheck.ScoreReset
    For Each bcCourt In bcCourtsToCheck
        '-- Award two points for every reporter clause that is compatible with the jurisdiction.
        bcCourt.Score = bcCourt.Score + 2 * ReportersCompatibleWithCourt(bcCourt, Citation.ReporterClauses, Citation.Year)
        '-- Award one point if the date clause's year is within the court's date range.
        If Citation.Year > 0 And bcCourt.YearStart <= Citation.Year And Citation.Year <= bcCourt.YearEnd Then bcCourt.Score = bcCourt.Score
+ 1
    Next bcCourt

    '-- Create a collection of courts with scores that meet or exceed the minimum score and with scores within one point of the
    ' highest score.
    Set colCourts = bcCourtsToCheck.ScoresBest(lngMinimumScore, 1)

    '-- Filter out courts from the same jurisdiction that have the same abbreviation.
    Set colUniqueCourts = New Collection
    For Each bcCourt In colCourts
        bolIsUnique = True
        For Each bcUniqueCourt In colUniqueCourts
            If bcCourt.Parent Is bcUniqueCourt.Parent Then
                If bcCourt.AbrvName = bcUniqueCourt.AbrvName Then
                    bolIsUnique = False
                    Exit For
                End If
            End If
        Next bcUniqueCourt
        If bolIsUnique = True Then colUniqueCourts.Add bcCourt
    Next bcCourt

    Set CompatibleCourts = colUniqueCourts

End Function

```

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

```
Private mcolExpectedGroupCodes As Collection
Private mbolUseParallel As Boolean
Private mbolRulesDiffer As Boolean
Private mbolSubmittedToStateCourt As Boolean
```

```
Private Const InvalidPriority As Long = 1000
```

```
Public Sub CheckParallelReporters()
```

```
    Dim bcReporterClause As cClauseReporter
```

```
    If Not (Citation.Jurisdiction Is Nothing) Then
```

```
        If fOptions.ForCourt <> "Federal" And fOptions.ForCourt = Citation.Jurisdiction.FullName Then mbolSubmittedToStateCourt = True
```

```
        Call SetExpectedReporterGroups(mcolExpectedGroupCodes, mbolUseParallel, mbolRulesDiffer)
        Call CheckReporterRuleExceptions
```

```
        For Each bcReporterClause In Citation.ReporterClauses
```

```
            Call CheckUnrecognizedReporter(bcReporterClause)
            If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
            Call CheckRedundantReporter(bcReporterClause)
            If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
            Call CheckUnnecessaryReporter(bcReporterClause)
            If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
            Call CheckIncorrectOrder(bcReporterClause)
            If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
        Next bcReporterClause
```

```
        Call CheckPreferredReporterMissing
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
        Call CheckMissingReporter
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
        Call CheckMissingReporterGroup
        If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
```

```
    End If
```

```
End Sub
```

```
Private Sub SetExpectedReporterGroups(ByRef ExpectedReporterGroups As Collection, ByRef UseParallel As Boolean, ByRef RulesDiffer As Boolean)
```

```
    Return the correct group of expected reporters (parallel or non-parallel) for this citation, based on whether the document is being submitted to a state court within the same jurisdiction as the citation.
```

```
    If Not (Citation.Court Is Nothing) Then
```

```
        If Citation.Court.ParallelReporters.Count = 0 Then
            Set ExpectedReporterGroups = Citation.Court.NonParallelReporters
            UseParallel = False
            RulesDiffer = False
```

```
        ElseIf Citation.Court.NonParallelReporters.Count = 0 Then
            Set ExpectedReporterGroups = Citation.Court.ParallelReporters
            UseParallel = True
            RulesDiffer = False
```

```
        ElseIf mbolSubmittedToStateCourt Then
            Set ExpectedReporterGroups = Citation.Court.ParallelReporters
            UseParallel = True
            RulesDiffer = True
```

```
        Else
            Set ExpectedReporterGroups = Citation.Court.NonParallelReporters
            UseParallel = False
            RulesDiffer = True
```

```
    End If
```

```
End Sub
```

```
Private Sub CheckReporterRuleExceptions()
```

```
    If Not (Citation.DateClause Is Nothing) Then
```

```
        If Citation.Jurisdiction.FullName = "California" And Citation.DateClause.Year <= 1959 Then
```

```
            If Citation.Court.FullName = "Court of Appeal" Then
```

```
                Set mcolExpectedGroupCodes = New Collection
                If mbolSubmittedToStateCourt Then
                    mcolExpectedGroupCodes.Add 1
                    mcolExpectedGroupCodes.Add 2
                    mbolUseParallel = True
```

```
                Else
                    mcolExpectedGroupCodes.Add 2
                    mcolExpectedGroupCodes.Add 1
                    mbolUseParallel = False
```

```
            End If
```

```
        End If
```



```

If Citation.Court.FullName = "Appellate Departments of the Superior Court" Then
    Set mcolExpectedGroupCodes = New Collection
    If mcolSubmittedToStateCourt Then
        mcolExpectedGroupCodes.Add 1
        mcolExpectedGroupCodes.Add 2
        mcolUseParallel = True
    Else
        mcolExpectedGroupCodes.Add 2
        mcolExpectedGroupCodes.Add 1
        mcolUseParallel = False
    End If
End If

```

```

End If
End Sub

```

```

Private Sub CheckUnrecognizedReporter(CurrentClause As cClauseReporter)

```

```

'-- If a reporter is not recognized as used by this particular jurisdiction or court, then suggest deleting or changing the
' reporter clause.

```

```

Dim bcReporter As cReporter
Dim strSuggestion As String
Dim colPreferredReporters As Collection

```

```

If Not (Citation.Court Is Nothing) Then

```

```

    If Citation.ReporterClauses.Count > 1 Then

```

```

        '-- If the current clause's reporter is not used by the court and there is at least one other reporter clause,
        ' then suggest deleting the current clause.

```

```

        '>> Jones v. Smith, 100 F.3d 105, 110, 200 N.W.2d 205, 210 (2d Cir. 2000)
        '>> Jones v. Smith, 100 Minn. 105, 110, 200 N.W.2d 205, 210 (Ct. App. 1990)
        '>> Jones v. Smith, 100 N.W.2d 105, 110, 200 Ill. App. 205, 210 (1990)
        '>> Jones v. Smith, 100 F.2d 105, 110, 200 Ill. App. 205, 210 (1990)

```

```

        If Not (CurrentClause.Reporter.Parent Is Citation.Court.CourtGroup) Then

```

```

            Call ErrorForm.LoadError("P11 UnrecognizedReporter 1", CurrentClause.ClauseStart, CurrentClause.ClauseEnd, CurrentClause.Re
porter.FullName, "the " & Citation.Court.FullName)

```

```

            If Not ErrorForm.IgnoreError Then
                Call SuggestClauseDelete(CurrentClause)
                Call ErrorForm.Activate
            End If

```

```

        End If

```

```

    Else

```

```

        '-- If the current clause's reporter is not used by the court and it is the only reporter clause,
        ' then suggest changing it to a valid reporter.

```

```

        '>> Jones v. Smith, 200 N.W.2d 205, 210 (2d Cir. 2000)
        '>> Jones v. Smith, 100 Minn. 105, 110 (Ct. App. 1990)
        '>> Jones v. Smith, 100 U.S. 105, 110 (Minn. 1980)

```

```

        If Not (CurrentClause.Reporter.Parent Is Citation.Court.CourtGroup) Then

```

```

            Set colPreferredReporters = PreferredReporters()

```

```

            Call ErrorForm.LoadError("P11 UnrecognizedReporter 2", CurrentClause.ClauseStart, CurrentClause.ClauseEnd, CurrentClause.Re
porter.FullName, "the " & Citation.Court.FullName, ListOfReporters(colPreferredReporters))

```

```

            If Not ErrorForm.IgnoreError Then
                For Each bcReporter In colPreferredReporters
                    Call SuggestClauseChange(CurrentClause, bcReporter, CurrentClause.PinStart > 0)
                Next bcReporter
                Call ErrorForm.Activate
            End If

```

```

        End If

```

```

    End If

```

```

End If

```

```

End Sub

```

```

Private Sub CheckRedundantReporter(CurrentClause As cClauseReporter)

```

```

'-- If the citation contains another reporter with the same group code as the present reporter, then suggest that one
' be deleted.

```

```

'>> Jones v. Smith, 100 N.W. 105, 110, 200 N.W.2d 205, 210 (Minn. 1980)

```

```

Dim bcSubsequentReporterClause As cClauseReporter

```

```

If Citation.ReporterClauses.Count > 1 Then

```

```

    For Each bcSubsequentReporterClause In Citation.ReporterClauses

```

```

        If bcSubsequentReporterClause.ClauseStart > CurrentClause.ClauseStart Then

```

```

            If bcSubsequentReporterClause.Reporter.GroupCode = CurrentClause.Reporter.GroupCode Then

```

```

                If Not (bcSubsequentReporterClause.EditorClause Is CurrentClause Or CurrentClause.EditorClause Is bcSubsequentReporterC
lause) Then

```

```

                    Call ErrorForm.LoadError("P11 RedunantReporter", CurrentClause.ClauseStart, CurrentClause.ClauseEnd, bcSubsequentRe
porterClause.Reporter.AbrvName, CurrentClause.Reporter.AbrvName)

```

```

                    If Not ErrorForm.IgnoreError Then
                        Call SuggestClauseDelete(CurrentClause)

```

```

    Call SuggestClauseDelete(bcSubsequentReporterClause)
    Call ErrorForm.Activate
    If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
End If

```

```

End If
End If

```

```

End If
Next bcSubsequentReporterClause
End If

```

```
End Sub
```

```
Private Sub CheckUnnecessaryReporter(CurrentClause As cClauseReporter)
```

```

'-- If this reporter is used by the citation, but isn't necessary to satisfy the required reporters for this court, then suggest
'   deleting it.
'>> Jones v. Smith, 100 U.S. 105, 110, 200 S. Ct. 205, 210 (1980)
'>> Jones v. Smith, 100 Minn. 105, 110, 200 N.W.2d 205, 210 (1980) [not submitted to Minnesota court]

```

```

Dim bcPreferredReporter As cReporter
Dim strSuggestion As String

```

```

If Not (Citation.Court Is Nothing) Then
    If IsValidReporter(CurrentClause.Reporter) Then
        If Not HasNeededReporter(CurrentClause) Then

```

```

            If mbolRulesDiffer And Not mbolSubmittedToStateCourt Then

```

```

                Call ErrorForm.LoadError("P11 UnnecessaryReporter B", CurrentClause.ClauseStart, CurrentClause.ClauseEnd, AddArticle(Ci
tation.Jurisdiction.FullName), CurrentClause.Reporter.FullName, Citation.Court.FullName)
            Else

```

```

                Call ErrorForm.LoadError("P11 UnnecessaryReporter A", CurrentClause.ClauseStart, CurrentClause.ClauseEnd, CurrentClause
.Reporter.FullName, Citation.Court.FullName)
            End If

```

```

            If Not ErrorForm.IgnoreError Then
                Call SuggestClauseDelete(CurrentClause)
                Call ErrorForm.Activate
            End If

```

```

        End If

```

```

    End If

```

```
End Sub
```

```
Private Sub CheckIncorrectOrder(CurrentClause As cClauseReporter)
```

```

'-- If the current clause has a reporter that is in an incorrect position and should go earlier in the parallel citation, then
'   suggest transposing the clause into its correct position.
'>> Jones v. Smith, 200 N.W.2d 205, 210, 300 Minn. 305, 310 (1980)
'>> Jones v. Smith, 100 L. Ed 105, 110, 200 S. Ct. 205, 210, 300 U.S. 305, 310 (1980)

```

```

Dim bcReporterClauseAtExpectedPosition As cClauseReporter

```

```

If Citation.ReporterClauses.Count > 1 Then

```

```

    Set bcReporterClauseAtExpectedPosition = ClauseAtExpectedStart(CurrentClause.Reporter)
    If Not (bcReporterClauseAtExpectedPosition Is CurrentClause) Then

```

```

        Call ErrorForm.LoadError("P11 IncorrectOrder", CurrentClause.ClauseStart, CurrentClause.ClauseEnd, CurrentClause.Reporter.FullN
ame, bcReporterClauseAtExpectedPosition.Reporter.FullName)
    If Not ErrorForm.IgnoreError Then

```

```

        Call ErrorForm.Suggestion.Transpose(CurrentClause.ClauseStart, CurrentClause.ClauseEnd, bcReporterClauseAtExpectedPosition.
ClauseStart)
        Call ErrorForm.Activate
    End If

```

```

End If

```

```

End If
End Sub

```

```
End Sub
```

```
Private Sub CheckPreferredReporterMissing()
```

```

'-- If the citation should not have parallel reporters and the citation isn't using the best reporter, then suggest changing the
'   current highest-rated clause to the preferred reporter.
'>> Jones v. Smith, 100 S. Ct. 105, 110 (1980)
'>> Jones v. Smith, 100 L. Ed. 105, 110, 200 S. Ct. 205, 210 (1980)

```

```

Dim colPreferredReporters As Collection
Dim bcReporter As cReporter
Dim bcClauseToReplace As cClauseReporter

```

```

If mbolUseParallel = False And Citation.ReporterClauses.Count > 0 Then

```

```

    Set colPreferredReporters = PreferredReporters()
    If colPreferredReporters.Count > 0 Then

```

```

        Set bcClauseToReplace = Citation.ReporterClauses(1)

```

```

        Call ErrorForm.LoadError("P11 NotPreferredReporter", bcClauseToReplace.ClauseStart, bcClauseToReplace.ClauseEnd, ListOfReporter
s(colPreferredReporters), bcClauseToReplace.Reporter.FullName)
    If Not ErrorForm.IgnoreError Then

```

```

        For Each bcReporter In colPreferredReporters
            Call SuggestClauseChange(bcClauseToReplace, bcReporter, bcClauseToReplace.PinStart > 0)
        Next bcReporter

```

```

        Call ErrorForm.Activate
    End If

```

End If

End If

End If

End Sub

Private Sub CheckMissingReporter()

'-- If there are no reporter clauses, then suggest inserting a valid reporter clause.
'>> Jones v. Smith (Minn. 1970)

Dim colPreferredReporters As Collection
Dim bcReporter As cReporter
Dim strSuggestion As String

If Citation.ReporterClauses.Count = 0 Then

Set colPreferredReporters = PreferredReporters()
If colPreferredReporters.Count > 0 Then

Call ErrorForm.LoadError("Pl1 MissingReporter", 0, 0, ListOfReporters(colPreferredReporters))
If Not ErrorForm.IgnoreError Then
For Each bcReporter In colPreferredReporters
Call SuggestClauseInsert(bcReporter, False)
Next bcReporter
Call ErrorForm.Activate
End If

End If

End If

End Sub

Private Sub CheckMissingReporterGroup()

' If this citation should use parallel reporters, then check whether all of the necessary reporter groups are satisfied.
'>> Jones v. Smith, 100 N.W.2d 105, 100 (Minn. 1970) [for Minnesota court]

Dim colSuggestedReporters As Collection
Dim varGroupCode As Variant
Dim bcReporter As cReporter
Dim strSuggestion As String

If mbolUseParallel = True And Citation.ReporterClauses.Count > 0 Then

For Each varGroupCode In mcolExpectedGroupCodes
If ClauseMatchingGroup(varGroupCode) Is Nothing Then

Set colSuggestedReporters = ValidReporters(varGroupCode)
If colSuggestedReporters.Count > 0 Then

If mbolSubmittedToStateCourt Then
Call ErrorForm.LoadError("Pl1 MissingReporterGroup 1", 0, 0, AddArticle(Citation.Jurisdiction.FullName), ListOfReporters(colSuggestedReporters))
Else
Call ErrorForm.LoadError("Pl1 MissingReporterGroup 2", 0, 0, ListOfReporters(colSuggestedReporters))
End If
If Not ErrorForm.IgnoreError Then
For Each bcReporter In ValidReporters(varGroupCode)
Call SuggestClauseInsert(bcReporter, False)
Next bcReporter
Call ErrorForm.Activate
If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub
End If

End If

End If

Next varGroupCode

End If

End Sub

Private Function ClauseAtExpectedStart(Reporter As cReporter) As cClauseReporter

'-- Returns the clause which occupies the position at which the reporter should be located. If the reporter already exists
' in the citation, returns its value in the current citation unless it should belong earlier in the parallel citations.
' i.e., this will not suggest a change if it should be moved later in the citation.

Dim bcReporterClause As cClauseReporter
Dim lngReporterPriority As Long

If Citation.ReporterClauses.Count > 0 Then

lngReporterPriority = Priority(Reporter)

'-- Iterate through each reporter clause.

For Each bcReporterClause In Citation.ReporterClauses

'-- If the reporter clause's reporter is the supplier reporter, then stop and return this clause as being in the
' correct position.

If Reporter Is bcReporterClause.Reporter Then
Set ClauseAtExpectedStart = bcReporterClause
Exit Function

Else

'-- If the reporter has a higher priority than the reporter clause it encounters, then return this reporter clause as

```

    where the reporter should be sorted.
    If lngReporterPriority < Priority(bcReporterClause.Reporter) Then
        Set ClauseAtExpectedStart = bcReporterClause
        Exit Function
    End If
End If
Next bcReporterClause

End If

End Function

Private Function EndOfLastReporterClause() As Long

    Dim bcLastReporterClause As cClauseReporter

    Set bcLastReporterClause = Citation.ReporterClauses(Citation.ReporterClauses.Count)
    If ActiveDoc.Words(bcLastReporterClause.ClauseEnd + 1) = "," Then
        EndOfLastReporterClause = bcLastReporterClause.ClauseEnd + 2
    Else
        EndOfLastReporterClause = bcLastReporterClause.ClauseEnd + 1
    End If

End Function

Private Function Priority(Reporter As cReporter) As Long

    '-- Returns a value representing the order in which the reporter's group code falls within the expected groupcodes collection.
    ' A lower number means that the reporter has greater "priority."

    Dim i As Long
    Dim lngReporterGroupCode As Long

    '-- If the reporter is valid for the citation's court, then assign its group code to the long. Otherwise, exit
    ' the function and return zero.
    If IsValidReporter(Reporter) Then
        lngReporterGroupCode = Reporter.GroupCode
    Else
        Priority = InvalidPriority
        Exit Function
    End If

    Step through each group code in the expected group codes to determine where the reporter's group code falls.
    For i = 1 To mcolExpectedGroupCodes.Count
        If mcolExpectedGroupCodes(i) = lngReporterGroupCode Then
            Priority = i
            Exit Function
        End If
    Next i

    A valid group code was not located.
    Priority = InvalidPriority

End Function

Private Function IsValidReporter(Reporter As cReporter) As Boolean

    Returns "true" if the reporter is used by the citation's court.

    If Not (Citation.Court Is Nothing) Then
        If Reporter.Parent.Parent Is Citation.Court.Parent Then IsValidReporter = True
    End If

End Function

' *****
' Ensure that each clause is sorted, from lowest to highest,
' by reporter group number. If not in the appropriate position, suggest moving
' the clause to the appropriate position.
' *****

Private Sub SortClausesByReporterGroup()

    Dim i As Long
    Dim j As Long
    Dim objLaterReporterClause As cClauseReporter
    Dim objEarlierReporterClause As cClauseReporter

    For i = 1 To Citation.Clauses.ReporterClauses.Count

        Set objEarlierReporterClause = Citation.Clauses.ReporterClauses(i)
        If Not (objEarlierReporterClause.Reporter Is Nothing) Then

            For j = i + 1 To Citation.Clauses.ReporterClauses.Count
                Set objLaterReporterClause = Citation.Clauses.ReporterClauses(j)
                If Not (objLaterReporterClause.Reporter Is Nothing) Then

                    If objLaterReporterClause.Reporter.GroupCode < objEarlierReporterClause.Reporter.GroupCode Then
                        Call mDisplayError.Transpose("T.1 Change Clause Position", objLaterReporterClause.ClauseStart, objLaterReporterClause.ClauseEnd, objEarlierReporterClause.ClauseStart, objEarlierReporterClause.ClauseEnd, objEarlierReporterClause.ReporterFullName, objLaterReporterClause.ReporterFullName)
                        If gbolStartOver = True Then Exit Sub
                    End If
                End If
            Next j
        End If
    Next i

End Sub

```

```

'End Sub

Private Sub CheckDisfavoredReporters()

    Dim bcReporterClause As cClauseReporter
    Dim strReporterAbvName As String
    Dim strSuggested As String

    For Each bcReporterClause In Citation.Clauses.ReporterClauses
        strReporterAbvName = bcReporterClause.ReporterAbvName

        If Citation.Year >= 1845 _
            And (strReporterAbvName = "Monag." _
            Or strReporterAbvName = "Sadler" _
            Or strReporterAbvName = "Walk." _
            Or strReporterAbvName = "Pennyp.") Then
            If CitationUsesPinpoints Then
                strSuggested = "__ Pa. __, __"
            Else
                strSuggested = "__ Pa. __"
            End If
            If Citation.Words(bcReporterClause.ClauseEnd) = "," Then strSuggested = strSuggested & ","
            Suggestion.FormattedText = Citation.FormattedText
            Call Suggestion.ReplaceWords(bcReporterClause.ClauseStart, bcReporterClause.ClauseEnd, strSuggested)
            Call mDisplayError.Change("T.1 Use Preferred Reporter, Single", bcReporterClause.ClauseStart, bcReporterClause.ClauseEnd, "Pa.
", bcReporterClause.ReporterAbvName)
            If gbolStartOver = True Then Exit Sub
        End If

        If strReporterAbvName = "Tex. Sup. Ct. J." Then
            If CitationUsesPinpoints Then
                strSuggested = "__ S.W.2d __, __"
            Else
                strSuggested = "__ S.W.2d __"
            End If
            If Citation.Words(bcReporterClause.ClauseEnd) = "," Then strSuggested = strSuggested & ","
            Suggestion.FormattedText = Citation.FormattedText
            Call Suggestion.ReplaceWords(bcReporterClause.ClauseStart, bcReporterClause.ClauseEnd, strSuggested)
            Call mDisplayError.Change("T.1 Use Preferred Reporter, Single", bcReporterClause.ClauseStart, bcReporterClause.ClauseEnd, "S.W.
2d", bcReporterClause.ReporterAbvName)
            If gbolStartOver = True Then Exit Sub
        End If
    Next bcReporterClause

'End Sub

Private Sub SuggestClauseDelete(ReporterClause As cClauseReporter)

    Call ErrorForm.Suggestion.Delete(ReporterClause.ClauseStart, ReporterClause.ClauseEnd)
    Call FixCourtAndJurisdiction(ReporterToDelete:=ReporterClause.Reporter)
    Call ErrorForm.DisplaySuggestion

End Sub

Private Sub SuggestClauseChange(ReporterClause As cClauseReporter, NewReporter As cReporter, IncludePinCite As Boolean)

    Dim strSuggestion As String

    strSuggestion = "__ " & NewReporter.AbrvName & " __"
    If IncludePinCite Then strSuggestion = strSuggestion & ", __"
    Call ErrorForm.Suggestion.Change(ReporterClause.ClauseStart, ReporterClause.ClauseEnd, strSuggestion)
    Call FixCourtAndJurisdiction(ReporterToDelete:=ReporterClause.Reporter, ReporterToAdd:=NewReporter)
    Call ErrorForm.DisplaySuggestion

End Sub

Private Sub SuggestClauseInsert(Reporter As cReporter, IncludePinCite As Boolean)

    Dim strSuggestion As String
    Dim bcLastReporterClause As cClauseReporter
    Dim lngInsertionPos As Long

    strSuggestion = "__ " & Reporter.AbrvName & " __,"
    If IncludePinCite Then strSuggestion = strSuggestion & " __,"
    If Citation.ReporterClauses.Count > 0 Then
        Set bcLastReporterClause = Citation.ReporterClauses(Citation.ReporterClauses.Count)
        lngInsertionPos = bcLastReporterClause.ClauseEnd + 1
    ElseIf Not (Citation.CaseNameClause Is Nothing) Then
        lngInsertionPos = Citation.CaseNameClause.ClauseEnd + 1
    Else
        lngInsertionPos = Citation.AllClauses(1).ClauseEnd
    End If

    If ActiveDoc.Words(lngInsertionPos) = "," Then
        lngInsertionPos = lngInsertionPos + 1
    Else
        strSuggestion = ", " & strSuggestion
    End If

    Call ErrorForm.Suggestion.Insert(lngInsertionPos, strSuggestion)
    Call FixCourtAndJurisdiction(ReporterToAdd:=Reporter)
    Call ErrorForm.DisplaySuggestion

End Sub

Private Sub FixCourtAndJurisdiction(Optional ReporterToDelete As cReporter, Optional ReporterToAdd As cReporter)

    If Citation.JurisdictionClause Is Nothing Then

```

```

'-- Add a jurisdiction clause if necessary.
If NeedsJurisdictionClause(ReporterToDelete, ReporterToAdd) Then
    Call ErrorForm.Suggestion.Insert(AppropriateJurisdictionPosition(Citation), Citation.Jurisdiction.AbrvName)
End If
Else
'-- Delete the jurisdiction clause if necessary.
If Not NeedsJurisdictionClause(ReporterToDelete, ReporterToAdd) Then
    Call ErrorForm.Suggestion.Delete(Citation.JurisdictionClause.JurStart, Citation.JurisdictionClause.JurEnd)
End If
End If

If Citation.CourtClause Is Nothing Then
'-- Add a court clause if necessary.
If NeedsCourtClause(ReporterToDelete, ReporterToAdd) Then
    Call ErrorForm.Suggestion.Insert(AppropriateCourtPosition(Citation), Citation.Court.AbrvName)
End If
Else
'-- Delete the court clause if necessary.
If Not NeedsCourtClause(ReporterToDelete, ReporterToAdd) Then
    Call ErrorForm.Suggestion.Delete(Citation.CourtClause.CourtStart, Citation.CourtClause.CourtEnd)
End If
End If

End Sub

Private Function NeedsJurisdictionClause(Optional ReporterToDelete As cReporter, Optional ReporterToAdd As cReporter) As Boolean

    Dim bcReporterClause As cClauseReporter

    If Not Citation.Jurisdiction Is Nothing Then
        '-- Start by assuming that a jurisdiction clause is needed
        NeedsJurisdictionClause = True

        '-- If the jurisdiction is federal, then a jurisdiction clause is not needed.
        If Citation.Jurisdiction.FullName = "Federal" Then
            NeedsJurisdictionClause = False
            Exit Function
        End If

        '-- If a reporter being added identifies the jurisdiction, then a jurisdiction clause is not needed.
        If Not (ReporterToAdd Is Nothing) Then
            If ReporterToAdd.ImplicitJurisdiction Is Citation.Jurisdiction Then
                NeedsJurisdictionClause = False
                Exit Function
            End If
        End If

        '-- If the jurisdiction is implicit from an existing reporter clause, then a jurisdiction clause is not needed.
        For Each bcReporterClause In Citation.ReporterClauses
            '-- If a reporter is being deleted, then don't check whether this reporter identifies the jurisdiction.
            If Not (bcReporterClause.Reporter Is ReporterToDelete) Then
                If bcReporterClause.Reporter.ImplicitJurisdiction Is Citation.Jurisdiction Then
                    NeedsJurisdictionClause = False
                    Exit Function
                End If
            End If
        Next bcReporterClause
    End If

End Function

Private Function NeedsCourtClause(Optional ReporterToDelete As cReporter, Optional ReporterToAdd As cReporter) As Boolean

    Dim bcReporterClause As cClauseReporter

    If Not Citation.Court Is Nothing Then
        '-- Start by assuming that a court clause is needed
        NeedsCourtClause = True

        '-- If the court is one that doesn't need an abbreviation (i.e., a supreme court), then a court clause is not needed.
        If Citation.Court.AbrvName = "" Then
            NeedsCourtClause = False
            Exit Function
        End If

        '-- If a reporter being added identifies the court, then a court clause is not needed.
        If Not (ReporterToAdd Is Nothing) Then
            If ReporterToAdd.ImplicitCourt Is Citation.Court Then
                NeedsCourtClause = False
                Exit Function
            End If
        End If

        '-- If the court is implicit from an existing reporter clause, then a court clause is not needed.
        For Each bcReporterClause In Citation.ReporterClauses
            '-- If a reporter is being deleted, then don't check whether this reporter identifies the court.
            If Not (bcReporterClause.Reporter Is ReporterToDelete) Then
                If bcReporterClause.Reporter.ImplicitCourt Is Citation.Court Then
                    NeedsCourtClause = False
                    Exit Function
                End If
            End If
        Next bcReporterClause
    End If
End If

```

End Function

Private Function HasNeededReporter(ReporterClause As cClauseReporter) As Boolean

```

Dim varGroupCode As Variant
Dim lngReporterClauseGroup As Long
Dim lngPriority As Long
Dim bcReporterClause As cClauseReporter

```

If mbolUseParallel Then

```

'-- If this citation should use parallel citations, then determine whether this reporter's group code is needed to satisfy
' one of the the required group codes.

```

If IsValidReporter(ReporterClause.Reporter) Then lngReporterClauseGroup = ReporterClause.Reporter.GroupCode

If ReporterClause.Reporter.Parent.Parent Is Citation.Court.Parent Then

lngReporterClauseGroup = ReporterClause.Reporter.GroupCode

End If

For Each varGroupCode In mcolExpectedGroupCodes

If varGroupCode = lngReporterClauseGroup Then

HasNeededReporter = True

Exit Function

End If

Next varGroupCode

Else

```

'-- If this citation shouldn't use parallel citations, then determine whether there's another reporter clause that
' has a higher priority over the current clause.

```

lngPriority = Priority(ReporterClause.Reporter)

If lngPriority < InvalidPriority Then

HasNeededReporter = True

For Each bcReporterClause In Citation.ReporterClauses

If Not (ReporterClause Is bcReporterClause) Then

If Priority(bcReporterClause.Reporter) < lngPriority Then

HasNeededReporter = False

Exit Function

End If

End If

Next bcReporterClause

End If

End If

End Function

Private Function PreferredReporters() As Collection

```

'-- Return a collection of reporters in the reporter group with valid group codes that are higher in priority than those
' already in the collection

```

Dim colPreferredReporters As Collection

Dim lngBestPriority As Long

Dim lngPriority As Long

Dim bcReporterClause As cClauseReporter

Dim bcReporter As cReporter

```

'-- Determine the highest priority within the current reporter clauses.

```

lngBestPriority = InvalidPriority + 1

For Each bcReporterClause In Citation.ReporterClauses

lngPriority = Priority(bcReporterClause.Reporter)

If lngPriority < lngBestPriority Then

lngBestPriority = lngPriority

End If

Next bcReporterClause

```

'-- Check each valid reporter. Add the reporter if it has a higher priority than the current highest priority.

```

Set colPreferredReporters = New Collection

For Each bcReporter In ValidReporters()

If Priority(bcReporter) < lngBestPriority Then colPreferredReporters.Add bcReporter

Next bcReporter

Set PreferredReporters = colPreferredReporters

End Function

Private Function BestClause() As cClauseReporter

```

'-- Returns the reporter clause with the highest priority.

```

Dim bcReporterClause As cClauseReporter

Dim lngBestPriority As Long

Dim lngPriority As Long

For Each bcReporterClause In Citation.ReporterClauses

If Not (bcReporterClause Is BestClause) Then

lngPriority = Priority(bcReporterClause.Reporter)

If lngPriority < lngBestPriority Or lngBestPriority = 0 Then

lngBestPriority = lngPriority

Set BestClause = bcReporterClause

End If

End If

Next bcReporterClause

End Function

Private Function ClauseMatchingGroup(GroupName As Variant) As cClauseReporter

'-- Returns an existing reporter clause in the citation that matches the supplied group code.

Dim bcReporterClause As cClauseReporter

For Each bcReporterClause In Citation.ReporterClauses

If IsValidReporter(bcReporterClause.Reporter) Then

If bcReporterClause.Reporter.GroupCode = GroupCode Then

Set ClauseMatchingGroup = bcReporterClause

Exit Function

End If

End If

Next bcReporterClause

End Function

Private Function ValidReporters(Optional GroupCode As Variant) As Collection

'-- Returns a collection of reporters that are not editor reporters and that match the date for the citation. If a group code is supplied, then only match reporters that have the same group code.

Dim bcReporter As cReporter

Dim lngYear As Long

Dim colValidReporters As Collection

Set colValidReporters = New Collection

If Not (Citation.DateClause Is Nothing) Then lngYear = Citation.DateClause.Year

For Each bcReporter In Citation.Court.Reporters

If Not (bcReporter.IsEditorReporter) Then

If (bcReporter.YearStart <= lngYear And lngYear <= bcReporter.YearEnd) Or lngYear = 0 Then

If IsMissing(GroupCode) Then

colValidReporters.Add bcReporter

ElseIf bcReporter.GroupCode = GroupCode Then

colValidReporters.Add bcReporter

End If

End If

End If

Next bcReporter

Set ValidReporters = colValidReporters

End Function

Private Function ListOfReporters(Reporters As Collection) As String

'-- Based on the supplied collection of reporters, create a string containing, e.g., "[ReporterA], [ReporterB] or [ReporterC]."

Dim bcReporter As cReporter

Dim i As Long

For i = 1 To Reporters.Count

If i = 1 Then

ListOfReporters = Reporters(i).AbrvName

ElseIf i < Reporters.Count Then

ListOfReporters = ListOfReporters & ", " & Reporters(i).AbrvName

Else

If Reporters.Count < 3 Then

ListOfReporters = ListOfReporters & " or " & Reporters(i).AbrvName

Else

ListOfReporters = ListOfReporters & ", or " & Reporters(i).AbrvName

End If

End If

Next i

End Function

mCheckReporter - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Public Sub CheckReporter()

Dim bcReporterClause As cClauseReporter

For Each bcReporterClause In Citation.ReporterClauses

Call CheckIncorrectAbrev(bcReporterClause)

If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckMissingVolumeNumber(bcReporterClause)

If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckIncorrectVolumeNumber(bcReporterClause)

If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckMissingFirstPage(bcReporterClause)

If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckPunctuationAfterFirstPage(bcReporterClause)

If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckMissingPinpoint(bcReporterClause)

If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Call CheckInvalidYear(bcReporterClause)

If ErrorForm.Result = Change Or ErrorForm.Result = Cancel Then Exit Sub

Next bcReporterClause

End Sub

Private Sub CheckIncorrectAbrev(CurrentClause As cClauseReporter)

' If the reporter is not punctuated or spelled correctly, then suggest correcting it.

Jones v. Smith, 200 U.S.S. 500, 505 (1980)

Jones v. Smith, 200 FSupp 500, 505 (D. Md. 1980)

Jones v. Smith, 100 N.2d 105, 110 (1980)

Dim colSuggestions As Collection

Dim colUniqueReporters As Collection

Dim bcReporter As cReporter

If CurrentClause.Reporter.AbrvName <> CurrentClause.AbrvText Then

'-- Generate a collection containing all of the reporters with similar names.

Set colSuggestions = New Collection

If Not (Citation.Court Is Nothing) Then Set colSuggestions = Citation.Court.Reporters.Items(CurrentClause.AbrvText, True, True)

If colSuggestions.Count = 0 Then

If Not (Citation.Jurisdiction Is Nothing) Then Set colSuggestions = Citation.Jurisdiction.AllReporters.Items(CurrentClause.AbrvText, True, True)

End If

If colSuggestions.Count = 0 Then

Set colSuggestions = Jurisdictions.AllReporters.Items(CurrentClause.AbrvText, True, True)

End If

'-- colSuggestions may contain reporters with duplicate abbreviations. Filter out the duplicate abbreviations.

Set colUniqueReporters = New Collection

For Each bcReporter In colSuggestions

If Not ExistsInCollection(bcReporter, colUniqueReporters) Then colUniqueReporters.Add bcReporter

Next bcReporter

'-- Display the error

If colUniqueReporters.Count = 1 Then

Call ErrorForm.LoadError("Rep IncorrectAbrev 1", CurrentClause.AbrvStart, CurrentClause.AbrvEnd, CurrentClause.Reporter.FullName)

If Not ErrorForm.IgnoreError Then

Call ErrorForm.Suggestion.Change(CurrentClause.AbrvStart, CurrentClause.AbrvEnd, CurrentClause.Reporter.AbrvName)

Call ErrorForm.Activate

End If

Else

Call ErrorForm.LoadError("Rep IncorrectAbrev 2", CurrentClause.AbrvStart, CurrentClause.AbrvEnd)

If Not ErrorForm.IgnoreError Then

For Each bcReporter In colUniqueReporters

Call ErrorForm.Suggestion.Change(CurrentClause.AbrvStart, CurrentClause.AbrvEnd, bcReporter.AbrvName)

Call ErrorForm.DisplaySuggestion

Next bcReporter

Call ErrorForm.Activate

End If

End If

End Sub

Private Sub CheckMissingVolumeNumber(CurrentClause As cClauseReporter)

'-- If the reporter citation is missing a volume number, then suggest inserting a placeholder or the correct volume number.

```
Dim lngExpectedVolumeNumber As Long
```

```
If CurrentClause.VolumeNumberLocation = 0 Then
```

```
'-- If this clause has an associated editor clause with a valid volume number, or an associated editor clause that
' doesn't need a volume number, then calculate the correct volume number for this clause.
```

```
If Not (CurrentClause.EditorClause Is Nothing) Then
```

```
With CurrentClause.EditorClause
```

```
If .VolumeNumber >= .Reporter.VolumeStart And .VolumeNumber <= .Reporter.VolumeEnd Then
```

```
lngExpectedVolumeNumber = (.VolumeNumber - .Reporter.VolumeStart) + .Reporter.MainReporterVolumeFirst
```

```
ElseIf .VolumeNumber = 0 Then
```

```
If .Reporter.VolumeStart = .Reporter.VolumeEnd Then lngExpectedVolumeNumber = .Reporter.MainReporterVolumeFirst
```

```
End If
```

```
End With
```

```
End If
```

```
If lngExpectedVolumeNumber > 0 Then
```

```
'>> Jones v. Smith, U.S. (3 Wheat.) 500, 505 (1820)
```

```
'>> Jones v. Smith, Ky. (Sneed) 500, 505 (1803)
```

```
Call ErrorForm.LoadError("Rep MissingVolumeNumber A", CurrentClause.AbrvStart, CurrentClause.AbrvEnd, CurrentClause.Reporter.FullName)
```

```
If Not ErrorForm.IgnoreError Then
```

```
Call ErrorForm.Suggestion.Insert(CurrentClause.AbrvStart, lngExpectedVolumeNumber)
```

```
Call ErrorForm.Activate
```

```
End If
```

```
Else
```

```
'>> Jones v. Smith, U.S. 500, 505 (1990)
```

```
Call ErrorForm.LoadError("Rep MissingVolumeNumber B", CurrentClause.AbrvStart, CurrentClause.AbrvEnd, CurrentClause.Reporter.FullName)
```

```
If Not ErrorForm.IgnoreError Then
```

```
Call ErrorForm.Suggestion.Insert(CurrentClause.AbrvStart, "__")
```

```
Call ErrorForm.Activate
```

```
End If
```

```
End If
```

```
End If
```

```
End Sub
```

```
Private Sub CheckIncorrectVolumeNumber(CurrentClause As cClauseReporter)
```

```
If the volume number is outside of the correct range of volume numbers for the reporter, then suggest changing or
replacing the volume number.
```

```
Jones v. Smith, 70 Del. 200, 205 (1960)
```

```
If CurrentClause.VolumeNumber > 0 Then
```

```
If Not (CurrentClause.Reporter.VolumeStart <= CurrentClause.VolumeNumber _
And CurrentClause.VolumeNumber <= CurrentClause.Reporter.VolumeEnd) Then
```

```
Call ErrorForm.LoadError("Rep IncorrectVolumeNumber", CurrentClause.VolumeNumberLocation, CurrentClause.VolumeNumberLocation, CurrentClause.Reporter.FullName)
```

```
If Not ErrorForm.IgnoreError Then
```

```
Call ErrorForm.Suggestion.Change(CurrentClause.VolumeNumberLocation, CurrentClause.VolumeNumberLocation, "__")
```

```
Call ErrorForm.Activate
```

```
End If
```

```
End If
```

```
End If
```

```
End Sub
```

```
Private Sub CheckMissingFirstPage(CurrentClause As cClauseReporter)
```

```
'-- If the first page number is missing, then suggest inserting a placeholder.
```

```
'>> Jones v. Smith, 100 N.W.2d (Minn. 1990)
```

```
'>> Jones v. Smith, 100 N.W.2d, 205 (Minn. 1990)
```

```
If CurrentClause.FirstPageLocation = 0 Then
```

```
Call ErrorForm.LoadError("Rep MissingFirstPage", 0, 0, CurrentClause.Reporter.FullName)
```

```
If Not ErrorForm.IgnoreError Then
```

```
Call ErrorForm.Suggestion.Insert(CurrentClause.AbrvEnd + 1, "__")
```

```
Call ErrorForm.Activate
```

```
End If
```

```
End If
```

```
End Sub
```

```
Private Sub CheckPunctuationAfterFirstPage(CurrentClause As cClauseReporter)
```

```
'-- If the punctuation between the first page number and the pinpoint page number is not a single comma, then suggest inserting
' a comma or changing the existing punctuation to a comma.
```

```
Dim strPunctuation As String
```

```
If CurrentClause.FirstPageLocation > 0 _
And CurrentClause.PinStart > 0 Then
```

```
If CurrentClause.PinStart = CurrentClause.FirstPageLocation + 1 Then
```

```
'-- Suggest inserting a comma.
```

```
'>> Jones v. Smith, 123 N.W.2d 456 480 (Minn. 1980)
```

```

Call ErrorForm.LoadError("Rep PunctuationAfterFirstPage A", CurrentClause.FirstPageLocation, CurrentClause.PinStart)
If Not ErrorForm.IgnoreError Then
    Call ErrorForm.Suggestion.Insert(CurrentClause.PinStart, ",")
    Call ErrorForm.Activate
End If

```

```

Else
    strPunctuation = ActiveDoc.Words(CurrentClause.FirstPageLocation + 1, CurrentClause.PinStart - 1)
    If strPunctuation <> "," Then

```

```

'-- Suggest changing the incorrect punctuation to a comma.
'>> Jones v. Smith, 123 N.W.2d 456,, 480 (Minn. 1980)
'>> Jones v. Smith, 123 N.W.2d 456) 480 (Minn. 1980)

```

```

    Call ErrorForm.LoadError("Rep PunctuationAfterFirstPage B", CurrentClause.FirstPageLocation + 1, CurrentClause.PinStart - 1)

```

```

    If Not ErrorForm.IgnoreError Then
        Call ErrorForm.Suggestion.Change(CurrentClause.FirstPageLocation + 1, CurrentClause.PinStart - 1, ",")
        Call ErrorForm.Activate
    End If

```

```

End If

```

```

End If

```

```

End If

```

```

End Sub

```

```

Private Sub CheckMissingPinpoint(CurrentClause As cClauseReporter)

```

```

'-- If the pinpoint page number is missing, then suggest inserting a placeholder for the pinpoint page number.
'>> Jones v. Smith, 123 N.W.2d 456 (Minn. 1980)

```

```

If CurrentClause.PinStart = 0 And CurrentClause.FirstPageLocation > 0 Then

```

```

    Call ErrorForm.LoadError("Dec MissingPinpoint", 0, 0, CurrentClause.Reporter.FullName)

```

```

    If Not ErrorForm.IgnoreError Then

```

```

        If ActiveDoc.Words(CurrentClause.FirstPageLocation + 1) = "," Then

```

```

            Call ErrorForm.Suggestion.Insert(CurrentClause.FirstPageLocation + 2, "__,")

```

```

        Else

```

```

            Call ErrorForm.Suggestion.Insert(CurrentClause.FirstPageLocation + 1, ", __,")

```

```

        End If

```

```

        Call ErrorForm.Activate

```

```

    End If

```

```

End If

```

```

End Sub

```

```

Private Sub CheckInvalidYear(CurrentClause As cClauseReporter)

```

```

'-- If the associated reporter's year range does not correspond to the year in the date clause, then suggest that the user
correct the discrepancy.
'>> Jones v. Smith, 100 Minn. 105, 110, 200 N.W.2d 205, 210 (1990)

```

```

If Citation.Year > 0 And Not (Citation.Court Is Nothing) Then

```

```

'-- Do not check the date if the reporter is not used by this court.

```

```

If CurrentClause.Reporter.Parent.Parent Is Citation.Court.Parent Then

```

```

    If Citation.Year < CurrentClause.Reporter.YearStart _
    Or Citation.Year > CurrentClause.Reporter.YearEnd Then

```

```

'-- Only flag an error if the year is within court's year range. (Otherwise, the error will be handled elsewhere.)
If Citation.Court.YearStart <= Citation.Year And Citation.Year <= Citation.Court.YearEnd Then

```

```

    Call ErrorForm.LoadError("Rep InvalidYear 1", Citation.DateClause.DateStart, Citation.DateClause.DateEnd, CurrentClause

```

```

.Reporter.AbrvName)
    If Not ErrorForm.IgnoreError Then

```

```

        Call ErrorForm.Suggestion.Change(Citation.DateClause.DateStart, Citation.DateClause.DateEnd, "____")

```

```

        Call ErrorForm.Activate

```

```

    End If

```

```

End If

```

```

End If

```

```

End If

```

```

End If

```

```

End Sub

```


mPublicFunctions - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Public Function ClausesAreAdjacent(ByRef FirstClause As IClause, ByRef SecondClause As IClause) As Boolean

 If FirstClause Is Nothing Or SecondClause Is Nothing Then Exit Function

 If ActiveDoc.Words.NextFullWord(FirstClause.ClauseEnd) >= SecondClause.ClauseStart Then
 ClausesAreAdjacent = True
 End If

End Function

Public Function AppropriateJurisdictionPosition(ByRef Citation As cCitation) As Long

 '-- Returns the word number of where the jurisdiction should be in the citation.

 AppropriateJurisdictionPosition = AppropriateDecisionClausePosition(Citation)
 If ActiveDoc.Words(AppropriateJurisdictionPosition) = "(" Then
 AppropriateJurisdictionPosition = AppropriateJurisdictionPosition + 1
 End If

End Function

Public Function AppropriateCourtPosition(ByRef Citation As cCitation) As Long

 '-- Returns the word number of where the court should be in the citation.

 If Citation.JurisdictionClause Is Nothing Then
 AppropriateCourtPosition = AppropriateJurisdictionPosition(Citation)
 Else
 AppropriateCourtPosition = Citation.JurisdictionClause.JurEnd + 1
 End If

End Function

Public Function AppropriateDatePosition(ByRef Citation As cCitation) As Long

 If Citation.JurisdictionClause Is Nothing _
 And Citation.CourtClause Is Nothing Then

 AppropriateDatePosition = AppropriateJurisdictionPosition(Citation)

 ElseIf Not (Citation.CourtClause Is Nothing) Then

 If Not (Citation.DateClause Is Nothing) Then
 If ActiveDoc.Words.NextFullWord(Citation.CourtClause.CourtStart) = Citation.DateClause.DateStart Then
 AppropriateDatePosition = Citation.DateClause.DateStart
 Else
 AppropriateDatePosition = Citation.CourtClause.CourtEnd + 1
 End If
 Else
 AppropriateDatePosition = Citation.CourtClause.CourtEnd + 1
 End If

 ElseIf Not (Citation.JurisdictionClause Is Nothing) Then

 If Not (Citation.DateClause Is Nothing) Then
 If ActiveDoc.Words.NextFullWord(Citation.JurisdictionClause.JurEnd) = Citation.DateClause.DateStart Then
 AppropriateDatePosition = Citation.DateClause.DateStart
 Else
 AppropriateDatePosition = Citation.JurisdictionClause.JurEnd + 1
 End If
 Else
 AppropriateDatePosition = Citation.JurisdictionClause.JurEnd + 1
 End If

 End If

End Function

Public Function AppropriateDecisionClausePosition(ByRef Citation As cCitation) As Long

 '-- Returns the word number of where the decision clause should begin in the citation.

 Dim bcClause As IClause

 Set bcClause = LastClauseOfType(Citation, "cClauseReporter")

 If Not (bcClause Is Nothing) Then

 AppropriateDecisionClausePosition = bcClause.ClauseEnd + 1

 Else

 Set bcClause = LastClauseOfType(Citation, "cClauseJurisdiction")

 If Not (bcClause Is Nothing) Then

 AppropriateDecisionClausePosition = bcClause.ClauseStart

 Else

 Set bcClause = LastClauseOfType(Citation, "cClauseCourt")

 If Not (bcClause Is Nothing) Then

 AppropriateDecisionClausePosition = bcClause.ClauseStart

 Else

 Set bcClause = LastClauseOfType(Citation, "cClauseDate")

 If Not (bcClause Is Nothing) Then

 AppropriateDecisionClausePosition = bcClause.ClauseStart

 Else

 AppropriateDecisionClausePosition = Citation.CitationEnd

 End If

 End If

 End If

 End If

```
If ActiveDoc.Words(AppropriateDecisionClausePosition) = "," _  
And Not (AppropriateDecisionClausePosition = Citation.CitationEnd) Then  
    AppropriateDecisionClausePosition = AppropriateDecisionClausePosition + 1  
End If
```

End Function

Private Function LastClauseOfType(Citation As cCitation, ClauseTypeName As String) As IClause

```
Dim bcClause As IClause  
Dim i As Long
```

```
For i = Citation.AllClauses.Count To 1 Step -1  
    Set bcClause = Citation.AllClauses(i)  
    If TypeName(bcClause) = ClauseTypeName Then  
        Set LastClauseOfType = bcClause  
        Exit Function  
    End If  
Next i
```

End Function

```
Public Function LastClauseOfType(Citation As cCitation, ClauseTypeName As String) As IClause  
    Dim bcClause As IClause  
    Dim i As Long  
    For i = Citation.AllClauses.Count To 1 Step -1  
        Set bcClause = Citation.AllClauses(i)  
        If TypeName(bcClause) = ClauseTypeName Then  
            Set LastClauseOfType = bcClause  
            Exit Function  
        End If  
    Next i  
End Function
```

```
'-- Copyright 2001 by Robert L. Jacobson
```

```
Option Explicit
```

```
Private mcolAllClauses As cClauses
```

```
Private mcolCaseNameClauses As Collection
```

```
Private mcolReporterClauses As Collection
```

```
Private mcolCourtClauses As Collection
```

```
Private mcolJurisdictionClauses As Collection
```

```
Private mbcCaseNameClause As cClauseCaseName
```

```
Private mbcCourtClause As cClauseCourt
```

```
Private mbcDateClause As cClauseDate
```

```
Private mbcJurisdictionClause As cClauseJurisdiction
```

```
Private mbcCourt As cCourt
```

```
Private mbcJurisdiction As cJurisdiction
```

```
Private mlngCitationStart As Long
```

```
Private mlngCitationEnd As Long
```

```
Private mlngCitationMaximumEnd As Long
```

```
Private mlngDecisionClauseStart As Long
```

```
Private mlngDecisionClauseEnd As Long
```

```
Private mbolWordUsed() As Boolean
```

```
Private mlngScore As Long
```

```
Public Property Get Court() As cCourt
```

```
Set Court = mbcCourt
```

```
End Property
```

```
Public Property Get Jurisdiction() As cJurisdiction
```

```
Set Jurisdiction = mbcJurisdiction
```

```
End Property
```

```
Public Property Get CitationStart() As Long
```

```
CitationStart = mlngCitationStart
```

```
End Property
```

```
Public Property Get CitationEnd() As Long
```

```
CitationEnd = mlngCitationEnd
```

```
End Property
```

```
Public Property Get AllClauses() As cClauses
```

```
Set AllClauses = mcolAllClauses
```

```
End Property
```

```
Public Property Get CaseNameClause() As cClauseCaseName
```

```
Set CaseNameClause = mbcCaseNameClause
```

```
End Property
```

```
Public Property Get CourtClause() As cClauseCourt
```

```
Set CourtClause = mbcCourtClause
```

```
End Property
```

```
Public Property Get DateClause() As cClauseDate
```

```
Set DateClause = mbcDateClause
```

```
End Property
```

```
Public Property Get JurisdictionClause() As cClauseJurisdiction
```

```
Set JurisdictionClause = mbcJurisdictionClause
```

```
End Property
```

```
Public Property Get ReporterClauses() As Collection
```

```
Set ReporterClauses = mcolReporterClauses
```

```
End Property
```

```
Public Property Get DecisionClauseStart() As Long
```

```
DecisionClauseStart = mlngDecisionClauseStart
```

```
End Property
```

```
Public Property Get DecisionClauseEnd() As Long
```

```
DecisionClauseEnd = mlngDecisionClauseEnd
```

```
End Property
```

```
Public Property Get WordIsRecognized(WordNumber As Long) As Boolean
```

```
WordIsRecognized = mbolWordUsed(WordNumber)
```

```
End Property
```

```
Public Property Get Text() As String
```

```
Text = ActiveDoc.Words(mlngCitationStart, mlngCitationEnd)
```

```
End Property
```

```
Friend Property Get Score() As Long
```

```
Score = mlngScore
```

```
End Property
```

```
Public Function Year() As Long
```

```
If Not (mbcDateClause Is Nothing) Then Year = mbcDateClause.Year
```

```
End Function
```

```
Public Sub Initialize(Clauses As Collection, CitationEnd)
```

```
Dim bcClause As IClause
```

```
'-- Identify the start and provisional (maximum) end of the citation.
```

```
Set bcClause = Clauses(1)
```

```
mlngCitationStart = bcClause.ClauseStart
```

```

If TypeOf bcClause Is cClauseJurisdiction Or TypeOf bcClause Is cClauseCourt Or TypeOf bcClause Is cClauseDate Then
    If ActiveDoc.Words(mlngCitationStart) = "(" Then mlngCitationStart = mlngCitationStart + 1
End If
mlngCitationEnd = CitationEnd

Call LoadClauses(Clauses)
Call MarkRecognizedWords

Set mbcJurisdiction = BestJurisdiction()
Set mbcCourt = BestCourt()

Call LocatePageAndVolumeNumbers
Call LocateCitationEnd

mlngScore = CalculateScore()

Call AssociateEditorReporters
Call AssociateReportersWithCourt

End Sub

Private Sub LoadClauses(Clauses As Collection)

    '-- Sort each clause into the respective collection.

    Dim bcClause As IClause
    Dim bcPriorReporterClause As cClauseReporter
    Dim bcEditor As cEditor

    Set mcolAllClauses = New cClauses
    Set mcolReporterClauses = New Collection
    Set mcolCaseNameClauses = New Collection
    Set mcolCourtClauses = New Collection
    Set mcolJurisdictionClauses = New Collection

    For Each bcClause In Clauses
        If TypeOf bcClause Is cClauseCaseName Then
            Set mbcCaseNameClause = bcClause
            mcolAllClauses.Add bcClause
            mcolCaseNameClauses.Add bcClause
        ElseIf TypeOf bcClause Is cClauseCourt Then
            mcolCourtClauses.Add bcClause
            mcolAllClauses.Add bcClause
        ElseIf TypeOf bcClause Is cClauseDate Then
            Set mbcDateClause = bcClause
            mcolAllClauses.Add bcClause
        ElseIf TypeOf bcClause Is cClauseJurisdiction Then
            mcolJurisdictionClauses.Add bcClause
            mcolAllClauses.Add bcClause
        ElseIf TypeOf bcClause Is cClauseReporter Then
            Set bcEditor = IsEditor(bcClause)
            If Not (bcEditor Is Nothing) Then
                Set bcPriorReporterClause = mcolAllClauses(mcolAllClauses.Count)
                Set bcPriorReporterClause.EditorClause = bcEditor
            Else
                mcolReporterClauses.Add bcClause
                mcolAllClauses.Add bcClause
            End If
            Set bcEditor = Nothing
        End If
    Next bcClause

End Sub

Private Function IsEditor(ReporterClause As cClauseReporter) As cEditor

    '-- If the last clause before this clause was a reporter clause, then test whether this clause looks sufficiently
    ' like an editor that it should be added as an editor to the prior reporter clause.

    Dim bcEditor As cEditor
    Dim bcPriorReporterClause As cClauseReporter

    If mcolReporterClauses.Count > 0 Then
        If TypeOf mcolAllClauses(mcolAllClauses.Count) Is cClauseReporter Then
            Set bcPriorReporterClause = mcolAllClauses(mcolAllClauses.Count)
            Set bcEditor = New cEditor
            With bcEditor
                Set .Reporter = ReporterClause.Reporter
                .AbrrStart = ReporterClause.AbrvStart
                .AbrrEnd = ReporterClause.AbrvEnd
                If .EditorScore(bcPriorReporterClause.AbrvEnd) >= 6 Then Set IsEditor = bcEditor
            End With
        End If
    End If

End Function

Private Function BestJurisdiction() As cJurisdiction

    '-- Sets mbcJurisdiction to the best jurisdiction indicated by the citation's jurisdiction clause, court clause and
    ' reporter clauses.

    Dim bcJurisdictionClause As cClauseJurisdiction

```



```

Dim bcReporterClause As cClauseReporter
Dim bcCourtClause As cClauseCourt
Dim bcJurisdiction As cJurisdiction
Dim strClauseText As String
Dim lngBestScore As Long

```

```
Call Jurisdictions.ScoreReset
```

```
For Each bcJurisdictionClause In mcolJurisdictionClauses
```

```

    '-- Award points for the jurisdiction identified by the jurisdiction clause(s).
    Set bcJurisdiction = bcJurisdictionClause.Jurisdiction
    bcJurisdiction.Score = bcJurisdiction.Score + 50 * bcJurisdictionClause.Score

```

```

    '-- Award points if, for each reporter clause, the reporter is used by this jurisdiction.
    For Each bcReporterClause In mcolReporterClauses
        If ReporterUsedByJurisdiction(bcReporterClause, bcJurisdiction) Then
            bcJurisdiction.Score = bcJurisdiction.Score + 20 * bcReporterClause.Score
        End If
    Next bcReporterClause

```

```
Next bcJurisdictionClause
```

```

'-- Award points for the jurisdiction implicit in any of the reporter clauses. Award three points if the jurisdiction is
' implicit in the court (e.g., a federal court); otherwise, award two points.

```

```

For Each bcReporterClause In mcolReporterClauses
    If bcReporterClause.Reporter.JurisdictionImplicit = True Then
        Set bcJurisdiction = bcReporterClause.Reporter.Parent.Parent
        If Not (bcReporterClause.Reporter.ImplicitCourt Is Nothing) Then
            bcJurisdiction.Score = bcJurisdiction.Score + 30 * bcReporterClause.Score
        Else
            bcJurisdiction.Score = bcJurisdiction.Score + 20 * bcReporterClause.Score
        End If
    End If
End If
Next bcReporterClause

```

```
For Each bcCourtClause In mcolCourtClauses
```

```
    If bcCourtClause.Court.JurisdictionImplicit = True Then
```

```

        '-- Award points for the jurisdiction implicit in the court clause(s).
        Set bcJurisdiction = bcCourtClause.Court.Parent
        bcJurisdiction.Score = bcJurisdiction.Score + 20 * bcCourtClause.Score
    
```

```
        '-- Award points if, for each reporter clause, the reporter is used by this jurisdiction.
```

```

        For Each bcReporterClause In mcolReporterClauses
            If ReporterUsedByJurisdiction(bcReporterClause, bcJurisdiction) Then
                bcJurisdiction.Score = bcJurisdiction.Score + 20 * bcReporterClause.Score
            End If
        Next bcReporterClause
    
```

```
End If
```

```
Next bcCourtClause
```

```

Select the jurisdiction with the highest score.
Set BestJurisdiction = Jurisdictions.ScoreBest

```

```
Search for the most appropriate jurisdiction clause, if there is one.
```

```
First, if the jurisdiction was identified above, then look for the jurisdiction clause that best matches this jurisdiction.
```

```

If Not (mbcJurisdiction Is Nothing) Then
    For Each bcJurisdictionClause In mcolJurisdictionClauses
        If bcJurisdictionClause.Jurisdiction Is mbcJurisdiction Then
            If mbcJurisdictionClause Is Nothing Then
                Set mbcJurisdictionClause = bcJurisdictionClause
                lngBestScore = bcJurisdictionClause.Score
            ElseIf bcJurisdictionClause.Score > lngBestScore Then
                Set mbcJurisdictionClause = bcJurisdictionClause
                lngBestScore = bcJurisdictionClause.Score
            End If
        End If
    Next bcJurisdictionClause
End If

```

```
Next bcJurisdictionClause
```

```
End If
```

```

'-- If no jurisdiction clause was identified in the first pass, just look for the best jurisdiction clause period (regardless of a
' matching court.)

```

```

If mbcJurisdictionClause Is Nothing Then
    For Each bcJurisdictionClause In mcolJurisdictionClauses
        If mbcJurisdictionClause Is Nothing Then
            Set mbcJurisdictionClause = bcJurisdictionClause
            lngBestScore = bcJurisdictionClause.Score
        ElseIf bcJurisdictionClause.Score > lngBestScore Then
            Set mbcJurisdictionClause = bcJurisdictionClause
            lngBestScore = bcJurisdictionClause.Score
        End If
    Next bcJurisdictionClause
End If

```

```
End Function
```

```
Private Function ReporterUsedByJurisdiction(ReporterClause As cClauseReporter, Jurisdiction As cJurisdiction) As Boolean
```

```
    Dim strReporterText As String
```

```
    strReporterText = ActiveDoc.Words(ReporterClause.AbrvStart, ReporterClause.AbrvEnd)
```

```
    If Not (Jurisdiction.AllReporters.Item(strReporterText, True, True) Is Nothing) Then ReporterUsedByJurisdiction = True
```

```
End Function
```

```
Private Function BestCourt() As cCourt
```

```
'-- Returns the best court indicated by the citation's clauses.
```

```
Dim lngYear As Long
Dim bcCourtClause As cClauseCourt
Dim bcReporterClause As cClauseReporter
Dim strClauseText As String
Dim bcFunctions As New cPublicFunctions
Dim bcCourt As cCourt
Dim lngBestScore As Long
```

```
If Not (mbcJurisdiction Is Nothing) Then
```

```
'-- If the jurisdiction was identified, then look for the clause that best identifies the court.
```

```
If Not mbcDateClause Is Nothing Then lngYear = mbcDateClause.Year
Call mbcJurisdiction.Courts.ScoreReset
```

```
'-- Award points for each court identified by a court clause that is consistent with the jurisdiction. Also, fix the
' court clause's associated court so that in "Minn. Ct. App.", e.g., the associated court refers to Minnesota's
' court of appeals, not the Alabama court of appeals.
```

```
For Each bcCourtClause In mcolCourtClauses
    strClauseText = bcCourtClause.Text
    Set bcCourt = mbcJurisdiction.Courts.Item(strClauseText, True, True, lngYear)
    If Not (bcCourt Is Nothing) Then
        Set bcCourtClause.Court = bcCourt
        bcCourt.Score = bcCourt.Score + 30
    End If
Next bcCourtClause
```

```
'-- Award points for each court that is implicit within the reporter clause.
```

```
For Each bcReporterClause In mcolReporterClauses
    Set bcCourt = bcReporterClause.Reporter.ImplicitCourt
    If Not (bcCourt Is Nothing) Then
        If bcCourt.Parent Is mbcJurisdiction Then
            bcCourt.Score = bcCourt.Score + 1
        End If
    End If
Next bcReporterClause
```

```
Set BestCourt = mbcJurisdiction.Courts.ScoreBest
```

```
'-- If no potential courts were found above, then look for a court within the jurisdiction that does not need an explicit
' court designation. (I.e., a federal or state supreme court.)
```

```
If BestCourt Is Nothing And Not (mbcJurisdiction Is Nothing) Then
```

```
    Set bcCourt = mbcJurisdiction.Courts.Item("", False, False, lngYear)
    If Not (bcCourt Is Nothing) Then
```

```
        If mbcJurisdiction.FullName = "Federal" Then
```

```
            '-- If the jurisdiction is federal and there is no court clause, then set the court to the U.S. Supreme Court
            ' only if there are no reporters or at least one of the reporters is valid for the Supreme Court.
            ' (Do not accept, e.g., "123 F.2d 456 (1980)" as the Supreme Court.)
```

```
            If mcolReporterClauses.Count = 0 Then
```

```
                Set BestCourt = bcCourt
```

```
            ElseIf bcFunctions.ReportersCompatibleWithCourt(bcCourt, mcolReporterClauses) >= 1 Then
```

```
                Set BestCourt = bcCourt
```

```
            End If
```

```
        Else
```

```
            Set BestCourt = bcCourt
```

```
        End If
```

```
    End If
```

```
End If
```

```
End If
```

```
'-- Search for the most appropriate court clause, if there is one.
```

```
'-- First, if the court was identified above, then look for the court clause that best matches this court.
```

```
If Not (mbcCourt Is Nothing) Then
```

```
    For Each bcCourtClause In mcolCourtClauses
```

```
        If bcCourtClause.Court Is mbcCourt Then
```

```
            If mbcCourtClause Is Nothing Then
```

```
                Set mbcCourtClause = bcCourtClause
```

```
                lngBestScore = bcCourtClause.Score
```

```
            ElseIf bcCourtClause.Score > lngBestScore Then
```

```
                Set mbcCourtClause = bcCourtClause
```

```
                lngBestScore = bcCourtClause.Score
```

```
            End If
```

```
        End If
```

```
    Next bcCourtClause
```

```
End If
```

```
'-- If no court clause was identified in the first pass, just look for the best court clause period (regardless of a
' matching court.)
```

```
If mbcCourtClause Is Nothing Then
```

```
    For Each bcCourtClause In mcolCourtClauses
```

```
        If mbcCourtClause Is Nothing Then
```

```
            Set mbcCourtClause = bcCourtClause
```

```
            lngBestScore = bcCourtClause.Score
```

```
        ElseIf bcCourtClause.Score > lngBestScore Then
```

```
            Set mbcCourtClause = bcCourtClause
```

```
            lngBestScore = bcCourtClause.Score
```

```
        End If
```

```
    Next bcCourtClause
```

```
End If
```

End Function

Private Sub LocatePageAndVolumeNumbers()

```
'-- For each reporter clause, identify the volume number, first page number and pinpoint page numbers.
' Note: this first identifies the volume number, then the first page number, and then the volume number, in that order,
' so that if one reporter clause "wants" that number as a page number, but the other wants it as a volume number, the
' volume number will prevail.
```

```
Dim bcReporterClause As cClauseReporter
Dim lngWord As Long
Dim strWord As String
Dim strWordTrim As String
Dim i As Long
```

```
'-- Identify the volume numbers.
```

```
For Each bcReporterClause In mcolReporterClauses
    lngWord = bcReporterClause.ClauseStart - 1
    If Not mbolWordUsed(lngWord) Then
        If ActiveDoc.Words.IsNumber(lngWord, True) Then
            bcReporterClause.VolumeNumberLocation = lngWord
            mbolWordUsed(lngWord) = True
        End If
    End If
Next bcReporterClause
```

```
'-- Identify the first page locations.
```

```
For Each bcReporterClause In mcolReporterClauses
    lngWord = bcReporterClause.ClauseEnd + 1
    If ActiveDoc.Words.IsNumber(lngWord, True) Then
        bcReporterClause.FirstPageLocation = lngWord
        mbolWordUsed(lngWord) = True
    ElseIf ActiveDoc.Words(lngWord) = "." Then
        If Not mbolWordUsed(lngWord + 1) Then
            If ActiveDoc.Words.IsNumber(lngWord + 1, True) Then
                bcReporterClause.FirstPageLocation = lngWord + 1
                mbolWordUsed(lngWord + 1) = True
            End If
        End If
    End If
End If
```

```
Next bcReporterClause
```

```
'-- Identify the pinpoint page numbers
```

```
For Each bcReporterClause In mcolReporterClauses
    With bcReporterClause
```

```
        For i = bcReporterClause.ClauseEnd + 1 To mlngCitationEnd
```

```
            If mbolWordUsed(i) = True Then Exit For
```

```
            strWordTrim = ActiveDoc.Words.WordsTrim(i)
            strWord = ActiveDoc.Words(i)
```

```
            If strWord <> "," Then
```

```
                If ActiveDoc.Words.IsNumber(i, True) Then
                    If .PinStart = 0 Then .PinStart = i
                    bcReporterClause.PinEnd = i
```

```
                ElseIf ActiveDoc.Words.IsNumber(i + 1, True) Then
```

```
                    If strWordTrim = "page" _
                    Or strWordTrim = "pages" _
                    Or strWordTrim = "note" _
                    Or strWordTrim = "notes" _
                    Or strWordTrim = "footnote" _
                    Or strWordTrim = "footnotes" _
                    Or strWordTrim = "to" _
                    Or strWordTrim = "and" _
                    Or strWordTrim = "at" _
                    Or strWord = "" _
                    Or strWord = "" _
                    Or strWord = "" _
                    Or strWord = "" _
                    Or strWordTrim = "p" _
                    Or strWordTrim = "pg" _
                    Or strWordTrim = "pp" _
                    Or strWordTrim = "n" _
                    Or strWordTrim = "nn" _
                    Or strWordTrim = "ns" _
                    Or strWordTrim = "fn" _
                    Or strWordTrim = "fns" Then
                        If .PinStart = 0 Then .PinStart = i
                        .PinEnd = i + 1
```

```
                    End If
```

```
                ElseIf ActiveDoc.Words(i + 1) = "." And ActiveDoc.Words.IsNumber(i + 2, True) Then
```

```
                    If strWordTrim = "p" _
                    Or strWordTrim = "pg" _
                    Or strWordTrim = "pp" _
                    Or strWordTrim = "n" _
                    Or strWordTrim = "nn" _
                    Or strWordTrim = "ns" _
                    Or strWordTrim = "fn" _
                    Or strWordTrim = "fns" Then
                        If .PinStart = 0 Then .PinStart = i
                        .PinEnd = i + 2
```

```
                    End If
```

```
                End If
            If .PinStart = 0 And strWordTrim <> "" Then Exit For
```

End If

Next i

If .PinEnd > 0 Then

For i = .PinStart To .PinEnd
 mbolWordUsed(i) = True

Next i

End If

End With

Next bcReporterClause

End Sub

Private Sub LocateCitationEnd()

Dim bcLastClause As IClause
Dim i As Long
Dim strWord As String
Dim lngInsideParenth As Long
Dim lngBestParenthEnd As Long

Set bcLastClause = mcolAllClauses(mcolAllClauses.Count)

'-- First, set the citation end to any semicolon that follows the last clause.

For i = bcLastClause.ClauseEnd + 1 To mlngCitationEnd

If ActiveDoc.Words(i) = ";" Then

mlngCitationEnd = i - 1

Exit For

End If

Next i

'-- Next, determine whether there is a mismatched parentheses count -- that is, whether the citation should have a close parenthesis after the last clause. Start by counting the parentheses to the left of the last clause

For i = bcLastClause.ClauseStart - 1 To mlngCitationStart Step -1

strWord = ActiveDoc.Words(i)

If strWord = "(" Then

lngInsideParenth = lngInsideParenth + 1

ElseIf strWord = ")" Then

lngInsideParenth = lngInsideParenth - 1

End If

Next i

'If the parentheses are imbalanced, then look for the correct number of closed parentheses after the last clause.

If lngInsideParenth > 0 Then

For i = bcLastClause.ClauseEnd + 1 To mlngCitationEnd

strWord = ActiveDoc.Words(i)

If strWord = "(" Then

lngInsideParenth = lngInsideParenth + 1

ElseIf strWord = ")" Then

lngInsideParenth = lngInsideParenth - 1

lngBestParenthEnd = i

If lngInsideParenth = 0 Then Exit For

End If

Next i

If lngBestParenthEnd > 0 Then mlngCitationEnd = lngBestParenthEnd

End If

End Sub

Private Function CalculateScore() As Long

Dim lngScore As Long
Dim lngAverageClauseScore As Long
Dim bcClause As IClause
Dim bcJurisdictionClause As cClauseJurisdiction
Dim bcCourtClause As cClauseCourt
Dim bcReporterClause As cClauseReporter
Dim bcClauseIDingJur As IClause
Dim bcClauseIDingCourt As IClause
Dim i As Long

'-- Award up to 100 points for the case name clause.

If Not (mbcCaseNameClause Is Nothing) Then

Set bcClause = mbcCaseNameClause

lngScore = 100 * bcClause.Score

End If

'-- Award up to 150 points based on the average reporter clause score.

If mcolReporterClauses.Count > 0 Then

For Each bcClause In mcolReporterClauses

lngAverageClauseScore = lngAverageClauseScore + 150 * bcClause.Score

Next bcClause

lngAverageClauseScore = lngAverageClauseScore / mcolReporterClauses.Count

lngScore = lngScore + lngAverageClauseScore

End If

'-- Award up to 100 points for the identification of the jurisdiction.

If Not (mbcJurisdiction Is Nothing) Then

For Each bcJurisdictionClause In mcolJurisdictionClauses

If bcJurisdictionClause.Jurisdiction Is mbcJurisdiction Then

Set bcClauseIDingJur = bcJurisdictionClause

Exit For

End If

Next bcJurisdictionClause

If bcClauseIDingJur Is Nothing Then

For Each bcCourtClause In mcolCourtClauses

If bcCourtClause.Court.ImplicitJurisdiction Is mbcJurisdiction Then

```

        Set bcClauseIDingJur = bcClause
        Exit For
    End If
    Next bcCourtClause
End If
If bcClauseIDingJur Is Nothing Then
    For Each bcReporterClause In mcolReporterClauses
        If bcReporterClause.Reporter.ImplicitJurisdiction Is mbcJurisdiction Then
            Set bcClauseIDingJur = bcReporterClause
            Exit For
        End If
    Next bcReporterClause
End If
If Not (bcClauseIDingJur Is Nothing) Then lngScore = lngScore + 100 * bcClauseIDingJur.Score
End If

'-- Award up to 100 points for the clause that identifies the court. If the court was identified, award points based on the
' the clause that IDd it. Otherwise, award points for the identified court clause, if any.

If Not (mbcCourt Is Nothing) Then
    For Each bcCourtClause In mcolCourtClauses
        If bcCourtClause.Court Is mbcCourt Then
            Set bcClauseIDingCourt = bcCourtClause
            Exit For
        End If
    Next bcCourtClause
    If bcClauseIDingCourt Is Nothing Then
        For Each bcReporterClause In mcolReporterClauses
            If bcReporterClause.Reporter.ImplicitCourt Is mbcCourt Then
                Set bcClauseIDingCourt = bcReporterClause
                Exit For
            End If
        Next bcReporterClause
    End If
    If bcClauseIDingCourt Is Nothing Then
        If mbcCourt.AbrvNameTrim = "" Then
            Set bcClauseIDingCourt = bcClauseIDingJur
        End If
    End If
End If
If Not (bcClauseIDingCourt Is Nothing) Then
    lngScore = lngScore + 50 * bcClauseIDingCourt.Score
ElseIf Not (mbcCourtClause Is Nothing) Then
    Set bcClauseIDingCourt = mbcCourtClause
    lngScore = lngScore + 100 * bcClauseIDingCourt.Score
End If

Award up to 100 points for the identification of the date.
If Not (mbcDateClause Is Nothing) Then
    Set bcClause = mbcDateClause
    lngScore = lngScore + 100 * bcClause.Score
End If

'-- Subtract 50 points for each case name clause, court clause or jurisdiction clause greater than one.
If mcolCaseNameClauses.Count > 1 Then lngScore = lngScore - 50 * (mcolCaseNameClauses.Count - 1)
If mcolCourtClauses.Count > 1 Then lngScore = lngScore - 50 * (mcolCourtClauses.Count - 1)
If mcolJurisdictionClauses.Count > 1 Then lngScore = lngScore - 50 * (mcolJurisdictionClauses.Count - 1)

'-- Subtract 50 points for each instance where the reporter clauses are not contiguous.
If mcolReporterClauses.Count > 1 Then
    '-- Step through every reporter clause except the last one, and determine if its next clause is a reporter clause.
    For i = 1 To mcolReporterClauses.Count - 1
        Set bcClause = mcolReporterClauses(i)
        If Not (TypeOf bcClause.NextClause Is cClauseReporter) Then lngScore = lngScore - 50
    Next i
End If

'-- Subtract 10 points for each unrecognized word, and subtract 100 points if a semicolon is encountered.
For i = mlngCitationStart To mlngCitationEnd
    If Not mcolWordUsed(i) Then
        If Not (ActiveDoc.Words.WordsTrim(i) = "") Then lngScore = lngScore - 10
        If ActiveDoc.Words(i) = ";" Then lngScore = lngScore - 100
    End If
Next i

CalculateScore = lngScore

End Function

Friend Sub FinalizeCitation()

    Call MarkRecognizedWords
    If Not (mbcCourt Is Nothing) Then Call AssociateReportersWithCourt

End Sub

Private Sub AssociateReportersWithCourt()

    Dim bcReporterClause As cClauseReporter
    Dim bcReporter As cReporter
    Dim bcMatchingReporter As cReporter

    For Each bcReporterClause In mcolReporterClauses

        '-- If this particular reporter is not used by this court group, but there is a reporter with the same abbreviation
        ' used by this court group, then set the reporter clause's associated reporter to the correct reporter.

        Set bcReporter = bcReporterClause.Reporter
        If Not (bcReporter.Parent Is mbcCourt.Parent) Then

```

```

If ActiveDoc.Words.Trim(bcReporterClause.AbrvStart, bcReporterClause.AbrvEnd) = bcReporter.AbrvNameTrim Then
    Set bcMatchingReporter = Court.Reporters.Item(bcReporterClause.AbrvText, True, False)
    If bcMatchingReporter Is Nothing Then Set bcMatchingReporter = mbcCourt.Parent.AllReporters(bcReporterClause.AbrvText, True, False)
Else
    Set bcMatchingReporter = Court.Reporters.Item(bcReporterClause.AbrvText, True, True)
    If bcMatchingReporter Is Nothing Then Set bcMatchingReporter = Court.Parent.AllReporters.Item(bcReporterClause.AbrvText, True, True)
End If

If Not (bcMatchingReporter Is Nothing) Then Set bcReporterClause.Reporter = bcMatchingReporter

End If
Next bcReporterClause

End Sub

Private Sub AssociateEditorReporters()
    Dim bcReporterClause As cClauseReporter
    Dim bcNextReporterClause As cClauseReporter
    Dim bcEditorMainReporter As cReporter
    Dim bcMainReporterClause As cClauseReporter
    Dim i As Long

    '-- If an editor clause wasn't attached above, then search for any associated main reporter clause.
    For Each bcReporterClause In mcolReporterClauses
        If bcReporterClause.EditorClause Is Nothing Then
            If bcReporterClause.Reporter.IsEditorReporter Then
                Set bcEditorMainReporter = bcReporterClause.Reporter.MainReporter
                For Each bcMainReporterClause In mcolReporterClauses
                    If bcMainReporterClause.Reporter Is bcEditorMainReporter Then
                        Set bcMainReporterClause.EditorClause = bcReporterClause
                        Exit For
                    End If
                Next bcMainReporterClause
            End If
        End If
    Next bcReporterClause
End Sub

Private Sub MarkRecognizedWords()
    Dim bcClause As IClause
    Dim i As Long

    For Each bcClause In mcolAllClauses
        For i = bcClause.ClauseStart To bcClause.ClauseEnd
            mbolWordUsed(i) = True
        Next i
    Next bcClause
End Sub

Friend Sub PrintClauses(Number As Long, PrintType As Boolean, PrintScore As Boolean)
    Dim bcClause As IClause
    Dim lngAddFrom As Long
    Dim strCitation As String
    Dim strInfo As String
    Dim strMisc As String

    If PrintType = True Or PrintScore = True Then
        lngAddFrom = mlngCitationStart
        For Each bcClause In mcolAllClauses
            strInfo = " ["
            If PrintType = True Then strInfo = strInfo & Right$(TypeName(bcClause), Len(TypeName(bcClause)) - 7)
            If PrintScore = True Then
                If PrintType = True Then strInfo = strInfo & ": "
                strInfo = strInfo & Format(bcClause.Score, "0.00")
            End If
            strInfo = strInfo & "]"
            strCitation = strCitation & ActiveDoc.Words(lngAddFrom, bcClause.ClauseEnd) & strInfo
            lngAddFrom = bcClause.ClauseEnd + 1
        Next bcClause
        strCitation = strCitation & ActiveDoc.Words(lngAddFrom, mlngCitationEnd)
    Else
        strCitation = ActiveDoc.Words(mlngCitationStart, mlngCitationEnd)
    End If

    If Number > 0 Then
        Debug.Print "#" & Number & ": " & strCitation
    Else
        Debug.Print strCitation
    End If

    If mbcJurisdiction Is Nothing Then
        strMisc = "Jurisdiction: none; "
    Else

```

```
    strMisc = "Jurisdiction: " & mbcJurisdiction.FullName & "; "
End If

If mbcCourt Is Nothing Then
    strMisc = strMisc & "Court: none; "
Else
    strMisc = strMisc & "Court: " & mbcCourt.FullName & "; "
End If

strMisc = strMisc & "Score: " & mngScore
Debug.Print strMisc
Debug.Print

End Sub
```

```

    strMisc = "Jurisdiction: " & mbcJurisdiction.FullName & "; "
End If

If mbcCourt Is Nothing Then
    strMisc = strMisc & "Court: none; "
Else
    strMisc = strMisc & "Court: " & mbcCourt.FullName & "; "
End If

strMisc = strMisc & "Score: " & mngScore
Debug.Print strMisc
Debug.Print

End Sub
```

cCitationBuilder - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

```
Private mlngNodesCount As Long
Private mbcBestCitation As cCitation
Private mbolPrintClauses As Boolean
Private mlngCitationsBuiltSingle As Long
Private mlngCitationsBuiltTotal As Long
Private mbcClauseLocator As cClauseLocator
```

```
Private Const MINIMUM_VALID_CITATION As Long = 150
Private Const ACCEPTABLE_DIFFERENCE As Long = 500
Private Const MINIMUM_CONTINUE As Long = 100
Private Const MAXIMUM_DISTANCE As Long = 30
Private Const PERFECT_SCORE As Long = 550
```

Public Function NextCitationFromWord(Word As Long) As cCitation

```
Dim colClauses As Collection
Dim bcClause As IClause
Dim i As Long
Dim j As Long
```

```
mbolPrintClauses = True
```

```
Set mbcBestCitation = Nothing
mlngCitationsBuiltSingle = 0
mlngNodesCount = 0
```

```
For i = Word To ActiveDoc.Words.Count
    For Each bcClause In mbcClauseLocator.ClausesStartingAtWord(i)
        Set colClauses = New Collection
        colClauses.Add bcClause
        Call TraverseTree(colClauses, 0)
    Next bcClause
    If Not (mbcBestCitation Is Nothing) Then Exit For
Next i
If Not (mbcBestCitation Is Nothing) Then
    If mbcBestCitation.Score < PERFECT_SCORE Then
        For j = mbcBestCitation.CitationStart + 1 To mbcBestCitation.CitationEnd
            For Each bcClause In mbcClauseLocator.ClausesStartingAtWord(j)
                Set colClauses = New Collection
                colClauses.Add bcClause
                Call TraverseTree(colClauses, 0)
            Next bcClause
        Next j
    End If
    Set NextCitationFromWord = mbcBestCitation
    Call mbcBestCitation.PrintClauses(0, True, True)
    Call mbcBestCitation.FinalizeCitation
End If
```

End Function

Private Sub TraverseTree(ParentClauses As Collection, ParentScore As Long)

```
Dim bcCitation As cCitation
Dim bcLastClause As IClause
Dim bcClause As IClause
Dim colClauses As Collection
```

```
Set bcLastClause = ParentClauses(ParentClauses.Count)
For Each bcClause In Children(bcLastClause.ClauseEnd)
```

```
    If Not (mbcBestCitation Is Nothing) Then If mbcBestCitation.Score = PERFECT_SCORE Then Exit Sub
```

```
    Set colClauses = CloneCollection(ParentClauses)
    colClauses.Add bcClause
```

```
    Set bcCitation = BuildCitation(colClauses)
    If mbolPrintClauses Then Call bcCitation.PrintClauses(mlngNodesCount, True, True)
```

```
    If bcCitation.Score > MINIMUM_CONTINUE And bcCitation.Score > (ParentScore - 100) Then
```

```
        If mbcBestCitation Is Nothing Then
            If bcCitation.Score > MINIMUM_VALID_CITATION Then Set mbcBestCitation = bcCitation
        ElseIf bcCitation.Score > mbcBestCitation.Score Then
            Set mbcBestCitation = bcCitation
        End If
        Call TraverseTree(colClauses, bcCitation.Score)
    End If
```

```
End If
```

```
Next bcClause
```

End Sub

Private Function BuildCitation(Clauses As Collection) As cCitation

```
Dim colClauses As Collection
Dim bcLastClause As IClause
```



```

Dim lngNextClauseStart As Long

mLngNodesCount = mLngNodesCount + 1
mLngCitationsBuiltSingle = mLngCitationsBuiltSingle + 1
mLngCitationsBuiltTotal = mLngCitationsBuiltTotal + 1

'-- Create a citation with these clauses and evaluate the result.
Set bcLastClause = Clauses(Clauses.Count)
For lngNextClauseStart = bcLastClause.ClauseEnd + 1 To ActiveDoc.Words.Count
    If mbcClauseLocator.ClausesStartingAtWord(lngNextClauseStart).Count > 0 Then Exit For
Next lngNextClauseStart

Set BuildCitation = New cCitation
Call BuildCitation.Initialize(Clauses, lngNextClauseStart - 1)

End Function

Private Function Children(AfterWord As Long) As Collection

'-- Search from the end of the parent clause to the end of the search range. Return the collection of clauses at the
'    next word that is the start of clauses.

Dim i As Long
Dim j As Long
Dim bcClause As IClause
Dim colClausesAtWord As Collection
Dim lngShortestClauseEnd As Long
Dim lngLastWord As Long

lngLastWord = ActiveDoc.Words.Count
If AfterWord + MAXIMUM_DISTANCE < lngLastWord Then lngLastWord = AfterWord + MAXIMUM_DISTANCE

For i = AfterWord + 1 To lngLastWord
    Set colClausesAtWord = mbcClauseLocator.ClausesStartingAtWord(i)
    If colClausesAtWord.Count > 0 Then

        '-- Add the clauses at the current word to the children collection
        Set Children = CloneCollection(mbcClauseLocator.ClausesStartingAtWord(i))

        '-- Determine the end of the shortest clause.
        For Each bcClause In Children
            If lngShortestClauseEnd = 0 Then
                lngShortestClauseEnd = bcClause.ClauseEnd
            ElseIf bcClause.ClauseEnd < lngShortestClauseEnd Then
                lngShortestClauseEnd = bcClause.ClauseEnd
            End If
        Next bcClause

        '-- Add the clauses that start after the current word, but before the end of the shortest clause
        For j = i + 1 To lngShortestClauseEnd
            Set colClausesAtWord = mbcClauseLocator.ClausesStartingAtWord(j)
            For Each bcClause In colClausesAtWord
                Children.Add bcClause
            Next bcClause
        Next j

        Exit Function
    End If
Next i
If Children Is Nothing Then Set Children = New Collection

End Function

Private Function CloneCollection(Source As Collection) As Collection

Dim bcClause As IClause

Set CloneCollection = New Collection
For Each bcClause In Source
    CloneCollection.Add bcClause.Clone
Next bcClause

End Function

Private Sub Class_Initialize()

Set mbcClauseLocator = New cClauseLocator
Call mbcClauseLocator.Initialize

End Sub

```

cClauseCaseName - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit
Implements IClause

Private mlngSignalStart As Long
Private mlngSignalEnd As Long
Private mlngClauseStart As Long
Private mlngClauseEnd As Long
Private mlngVersusPosition As Long
Private mudtPartyName() As udtPartyName
Private mlngPartiesCount As Long

Private mbcNextClause As IClause
Private mbcPriorClause As IClause

Public Type udtPartyName
Start As Long
End As Long
End Type

Public Property Let ClauseStart(Source As Long)
mlngClauseStart = Source
If mlngClauseStart > 0 And mlngClauseEnd > 0 Then Call Parse
End Property

Public Property Get ClauseStart() As Long
ClauseStart = mlngClauseStart
End Property

Public Property Let ClauseEnd(Source As Long)
mlngClauseEnd = Source
If mlngClauseStart > 0 And mlngClauseEnd > 0 Then Call Parse
End Property

Public Property Get ClauseEnd() As Long
ClauseEnd = mlngClauseEnd
End Property

Public Property Get SignalStart() As Long
SignalStart = mlngSignalStart
End Property

Public Property Get SignalEnd() As Long
SignalEnd = mlngSignalEnd
End Property

Public Property Get VersusPosition() As Long
VersusPosition = mlngVersusPosition
End Property

Public Property Get Text() As String
Text = ActiveDoc.Words(mlngClauseStart, mlngClauseEnd)
End Property

Public Property Get PartyName(PartyNumber As Long) As udtPartyName
PartyName = mudtPartyName(PartyNumber)
End Property

Public Function PartyNameAtWord(WordNumber As Long) As udtPartyName
If mlngVersusPosition = 0 Or WordNumber < mlngVersusPosition Then
PartyNameAtWord = mudtPartyName(1)
Else
PartyNameAtWord = mudtPartyName(2)
End If
End Function

Public Property Get PartiesCount() As Long
PartiesCount = mlngPartiesCount
End Property

Private Property Get IClause_ClauseEnd() As Long
IClause_ClauseEnd = mlngClauseEnd
End Property

Private Property Get IClause_ClauseStart() As Long
IClause_ClauseStart = mlngClauseStart
End Property

Private Function IClause_Clone() As IClause

Dim bcClone As cClauseCaseName
Set bcClone = New cClauseCaseName
With bcClone
.ClauseStart = mlngClauseStart
.ClauseEnd = mlngClauseEnd
End With
Set IClause_Clone = bcClone

End Function

Private Property Set IClause_NextClause(RHS As IClause)
Set mbcNextClause = RHS
End Property

Private Property Get IClause_NextClause() As IClause
Set IClause_NextClause = mbcNextClause
End Property

```
Private Property Set IClause_PriorClause() As IClause
    Set mbcPriorClause = RHS
End Property
```

```
Private Property Get IClause_PriorClause() As IClause
    Set IClause_PriorClause = mbcPriorClause
End Property
```

```
Private Property Get IClause_Text() As String
    IClause_Text = Text()
End Property
```

```
Private Function IClause_Score() As Single
```

```
    Dim sngScoreContext As Single
    Dim strNextWord As String
```

```
    '-- The total score is based solely on the context score.
```

```
    '-- Award up to 0.5 points for the expected preceeding clause and text.
```

```
    If mbcPriorClause Is Nothing Then
```

```
        sngScoreContext = 0.5
```

```
    ElseIf ActiveDoc.Words(mudtPartyName(1).Start - 1) = "," Then
```

```
        sngScoreContext = 0.25
```

```
    End If
```

```
    '-- Award up to 0.5 points for the expected subsequent clause and text.
```

```
    If Not (mbcNextClause Is Nothing) Then If TypeOf mbcNextClause Is cClauseReporter Then sngScoreContext = sngScoreContext + 0.25
```

```
    strNextWord = ActiveDoc.Words(mlngClauseEnd + 1)
```

```
    Select Case strNextWord
```

```
        Case ","
```

```
            sngScoreContext = sngScoreContext + 0.25
```

```
        Case "("
```

```
            sngScoreContext = sngScoreContext + 0.125
```

```
    End Select
```

```
    IClause_Score = sngScoreContext
```

```
End Function
```

```
Friend Function Score() As Single
```

```
    Score = IClause_Score()
```

```
End Function
```

```
Private Sub Parse()
```

```
    Dim i As Long
```

```
    Dim lngSearchTo As Long
```

```
    Dim strPhraseTrim As String
```

```
    Dim strWordTrim As String
```

```
    Dim lngFirstPartyNameStart As Long
```

```
    If mlngClauseStart + 5 > mlngClauseEnd Then
        lngSearchTo = mlngClauseEnd
```

```
    Else
        lngSearchTo = mlngClauseStart + 10
```

```
    End If
```

```
    For i = lngSearchTo To mlngClauseStart Step -1
```

```
        '-- Query: Will WordsTrim return a word when the trailing word(s) is a punctuation mark? Check.
```

```
        strPhraseTrim = ActiveDoc.Words.WordsTrim(mlngClauseStart, i)
```

```
        If strPhraseTrim = "accord" _
```

```
        Or strPhraseTrim = "accordeg" _
```

```
        Or strPhraseTrim = "see" _
```

```
        Or strPhraseTrim = "seeeg" _
```

```
        Or strPhraseTrim = "seealso" _
```

```
        Or strPhraseTrim = "seealsoeg" _
```

```
        Or strPhraseTrim = "cf" _
```

```
        Or strPhraseTrim = "cfeg" _
```

```
        Or strPhraseTrim = "compare" _
```

```
        Or strPhraseTrim = "compareeg" _
```

```
        Or strPhraseTrim = "butsee" _
```

```
        Or strPhraseTrim = "butseeeg" _
```

```
        Or strPhraseTrim = "butcf" _
```

```
        Or strPhraseTrim = "butcfeg" _
```

```
        Or strPhraseTrim = "seegenerally" _
```

```
        Or strPhraseTrim = "seegenerallyeg" _
```

```
        Or strPhraseTrim = "contra" _
```

```
        Or strPhraseTrim = "contraeg" Then
```

```
            mlngSignalStart = mlngClauseStart
```

```
            If ActiveDoc.Words(i + 1) = "." Then
```

```
                mlngSignalEnd = i + 1
```

```
            Else
```

```
                mlngSignalEnd = i
```

```
            End If
```

```
            Exit For
```

```
        End If
```

```
    Next i
```

```
    '-- Look for the first designation of "versus."
```

```
    For i = mlngClauseStart To mlngClauseEnd
```

```
        strWordTrim = ActiveDoc.Words.WordsTrim(i)
```

```
        If strWordTrim = "v" Or strWordTrim = "vs" Or strWordTrim = "versus" Then
```


cClauseCourt - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit
Implements IClause

Public Court As cCourt

Private mlngCourtStart As Long
Private mlngCourtEnd As Long
Private mbcNextClause As IClause
Private mbcPriorClause As IClause
Private mbolExactMatch As Boolean

Friend Property Let ExactMatch(Source As Boolean)
 mbolExactMatch = Source
End Property

Public Property Get CourtStart() As Long
 CourtStart = mlngCourtStart
End Property

Friend Property Let CourtStart(Source As Long)
 mlngCourtStart = Source
End Property

Public Property Get CourtEnd() As Long
 CourtEnd = mlngCourtEnd
End Property

Friend Property Let CourtEnd(Source As Long)
 mlngCourtEnd = Source
End Property

Public Property Get Text() As String
 Text = ActiveDoc.Words(mlngCourtStart, mlngCourtEnd)
End Property

Public Property Get PriorClause() As IClause
 Set PriorClause = mbcPriorClause
End Property

Public Property Get NextClause() As IClause
 Set NextClause = mbcNextClause
End Property

Private Property Get IClause_ClauseEnd() As Long
 IClause_ClauseEnd = mlngCourtEnd
End Property

Private Property Get IClause_ClauseStart() As Long
 IClause_ClauseStart = mlngCourtStart
End Property

Private Function IClause_Clone() As IClause

```
Dim bcClone As cClauseCourt
Set bcClone = New cClauseCourt
With bcClone
    .CourtStart = mlngCourtStart
    .CourtEnd = mlngCourtEnd
    Set .Court = Court
    .ExactMatch = mbolExactMatch
End With
Set IClause_Clone = bcClone
```

End Function

Private Property Set IClause_NextClause(RHS As IClause)
 Set mbcNextClause = RHS
End Property

Private Property Get IClause_NextClause() As IClause
 Set IClause_NextClause = mbcNextClause
End Property

Private Property Set IClause_PriorClause(RHS As IClause)
 Set mbcPriorClause = RHS
End Property

Private Property Get IClause_PriorClause() As IClause
 Set IClause_PriorClause = mbcPriorClause
End Property

Private Property Get IClause_Text() As String
 IClause_Text = Text
End Property

Private Function IClause_Score() As Single

```
Dim sngScoreContext As Single
Dim sngScoreInternal As Single
Dim strPriorWord As String
```

'-- Award up to 1 point for the internal score if the court is spelled correctly.

```
If mbolExactMatch Then
    sngScoreInternal = 0.75
    If ActiveDoc.Words(mlngCourtStart, mlngCourtEnd) = Court.AbrvName Then sngScoreInternal = sngScoreInternal + 0.25
End If
```

```
-- Award up to .5 points for the context score for the expected preceding clause and text.
If Not (mbcPriorClause Is Nothing) Then
    If Not (mIngCourtStart = mbcPriorClause.ClauseEnd) Then strPriorWord = ActiveDoc.Words(mIngCourtStart - 1)
    If TypeOf mbcPriorClause Is cClauseJurisdiction Then
        sngScoreContext = sngScoreContext + 0.25
        Select Case strPriorWord
            Case ""
                sngScoreContext = sngScoreContext + 0.25
            Case ", "
                sngScoreContext = sngScoreContext + 0.125
        End Select
    Else
        If TypeOf mbcPriorClause Is cClauseReporter Then sngScoreContext = sngScoreContext + 0.25
        Select Case strPriorWord
            Case "("
                sngScoreContext = sngScoreContext + 0.25
            Case ", ", ""
                sngScoreContext = sngScoreContext + 0.125
        End Select
    End If
End If

'-- Award up to .5 points for the context score for the expected subsequent clause and text.
If Not (mbcNextClause Is Nothing) Then
    If TypeOf mbcNextClause Is cClauseDate Then
        sngScoreContext = sngScoreContext + 0.25
        If mIngCourtEnd + 1 = mbcNextClause.ClauseStart Then sngScoreContext = sngScoreContext + 0.25
    Else
        If ActiveDoc.Words(mIngCourtEnd + 1) = ")" Then sngScoreContext = sngScoreContext + 0.25
    End If
Else
    If ActiveDoc.Words(mIngCourtEnd + 1) = ")" Then sngScoreContext = sngScoreContext + 0.25
End If

'-- Average the internal and context scores, with 40% weight for the internal score and 60% weight for the context score.
IClause_Score = 0.4 * sngScoreInternal + 0.6 * sngScoreContext

End Function

Friend Function Score() As Single
    Score = IClause_Score()
End Function

Private Sub Class_Initialize()
    ClausesCount = ClausesCount + 1
    Debug.Print "Creating clause. Total: " & ClausesCount
End Sub

Private Sub Class_Terminate()
    ClausesCount = ClausesCount - 1
    Debug.Print "Terminating clause. Total: " & ClausesCount
End Sub
```

cClauseDate - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit
Implements IClause

Public FullDate As Date

Private mlngDateStart As Long
Private mlngDateEnd As Long
Private mbcNextClause As IClause
Private mbcPriorClause As IClause
Private mlngYear As Long

Public Property Get DateStart() As Long
DateStart = mlngDateStart
End Property

Friend Property Let DateStart(Source As Long)
mlngDateStart = Source
End Property

Public Property Get DateEnd() As Long
DateEnd = mlngDateEnd
End Property

Friend Property Let DateEnd(Source As Long)
mlngDateEnd = Source
End Property

Public Property Get Text() As String
Text = ActiveDoc.Words(mlngDateStart, mlngDateEnd)
End Property

Public Property Get PriorClause() As IClause
Set PriorClause = mbcPriorClause
End Property

Public Property Get NextClause() As IClause
Set NextClause = mbcNextClause
End Property

Public Property Get Year() As Long
Year = mlngYear
End Property

Friend Property Let Year(Source As Long)
mlngYear = Source
End Property

Private Property Get IClause_ClauseEnd() As Long
IClause_ClauseEnd = mlngDateEnd
End Property

Private Property Get IClause_ClauseStart() As Long
IClause_ClauseStart = mlngDateStart
End Property

Private Function IClause_Clone() As IClause

```
Dim bcClone As cClauseDate
Set bcClone = New cClauseDate
With bcClone
.DateStart = mlngDateStart
.DateEnd = mlngDateEnd
.FullDate = FullDate
.Year = mlngYear
End With
Set IClause_Clone = bcClone
```

End Function

Private Property Set IClause_NextClause(RHS As IClause)
Set mbcNextClause = RHS
End Property

Private Property Get IClause_NextClause() As IClause
Set IClause_NextClause = mbcNextClause
End Property

Private Property Set IClause_PriorClause(RHS As IClause)
Set mbcPriorClause = RHS
End Property

Private Property Get IClause_PriorClause() As IClause
Set IClause_PriorClause = mbcPriorClause
End Property

Private Property Get IClause_Text() As String
IClause_Text = Text
End Property

Private Function IClause_Score() As Single

```
Dim sngScoreContext As Single
Dim sngScoreInternal As Single
Dim strClauseText As String
Dim strNextWord As String
Dim lngYear As Long
```

```

'-- Award up to 1 point for the internal score if the court is spelled correctly
strClauseText = Text()
If mlngDateStart = mlngDateEnd Then
    '-- If the date clause is only one word long, then it is either a year or a placeholder. Award points if it is a valid
    ' year or a valid (4-6 length) placeholder.
    If IsNumeric(strClauseText) Then
        lngYear = Val(strClauseText)
        If lngYear > 1600 And lngYear < 2010 Then sngScoreInternal = 1
    Else
        If Len(strClauseText) > 4 And Len(strClauseText) < 7 Then
            sngScoreInternal = 1
        Else
            sngScoreInternal = 0.5
        End If
    End If
Else
    '-- If the date clause is more than one word long, it contains a month and date. Award points if it has a year within
    ' a valid range.
    lngYear = Val(Format$(strClauseText, "yyyy"))
    If lngYear > 1600 And lngYear < 2010 Then
        sngScoreInternal = 1
    Else
        sngScoreInternal = 0.5
    End If
End If

'-- Award up to .5 points for the context score for the expected preceeding clause and text.
If Not (mbcPriorClause Is Nothing) Then
    If TypeOf mbcPriorClause Is cClauseCourt Or TypeOf mbcPriorClause Is cClauseJurisdiction Then
        sngScoreContext = sngScoreContext + 0.25
        If mbcPriorClause.ClauseEnd + 1 = mlngDateStart Then sngScoreContext = sngScoreContext + 0.25
    ElseIf TypeOf mbcPriorClause Is cClauseReporter Then
        sngScoreContext = sngScoreContext + 0.25
        If ActiveDoc.Words(mlngDateStart - 1) = "(" Then sngScoreContext = sngScoreContext + 0.25
    End If
End If

'-- Award up to .5 points for the context score for the expected subsequent clause and text.
strNextWord = ActiveDoc.Words(mlngDateEnd + 1)
If mbcNextClause Is Nothing Then
    If strNextWord = ")" Then
        sngScoreContext = sngScoreContext + 0.5
    ElseIf strNextWord = "," Or strNextWord = ";" Then
        sngScoreContext = sngScoreContext + 0.25
    End If
Else
    If strNextWord = ")" Then
        sngScoreContext = sngScoreContext + 0.25
    ElseIf strNextWord = "," Or strNextWord = ";" Then
        sngScoreContext = sngScoreContext + 0.125
    End If
End If

'-- Average the internal and context scores, with 40% weight for the internal score and 60% weight for the context score.
IClause_Score = 0.4 * sngScoreInternal + 0.6 * sngScoreContext

End Function

Friend Function Score() As Single
    Score = IClause_Score()
End Function

Private Sub Class_Initialize()
    ClausesCount = ClausesCount + 1
    Debug.Print "Creating clause. Total: " & ClausesCount
End Sub

Private Sub Class_Terminate()
    ClausesCount = ClausesCount - 1
    Debug.Print "Terminating clause. Total: " & ClausesCount
End Sub

```


cClauseJurisdiction - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit
Implements IClause

Public Jurisdiction As cJurisdiction

Private mlngJurStart As Long
Private mlngJurEnd As Long
Private mbcNextClause As IClause
Private mbcPriorClause As IClause
Private mbolExactMatch As Boolean

Friend Property Let ExactMatch(Source As Boolean)
 mbolExactMatch = Source
End Property

Public Property Get JurStart() As Long
 JurStart = mlngJurStart
End Property

Friend Property Let JurStart(Source As Long)
 mlngJurStart = Source
End Property

Public Property Get JurEnd() As Long
 JurEnd = mlngJurEnd
End Property

Friend Property Let JurEnd(Source As Long)
 mlngJurEnd = Source
End Property

Public Property Get Text() As String
 Text = ActiveDoc.Words(mlngJurStart, mlngJurEnd)
End Property

Public Property Get PriorClause() As IClause
 Set PriorClause = mbcPriorClause
End Property

Public Property Get NextClause() As IClause
 Set NextClause = mbcNextClause
End Property

Private Property Get IClause_ClauseEnd() As Long
 IClause_ClauseEnd = mlngJurEnd
End Property

Private Property Get IClause_ClauseStart() As Long
 IClause_ClauseStart = mlngJurStart
End Property

Private Function IClause_Clone() As IClause
 Dim bcClone As cClauseJurisdiction
 Set bcClone = New cClauseJurisdiction
 With bcClone
 .JurStart = mlngJurStart
 .JurEnd = mlngJurEnd
 .Set .Jurisdiction = Jurisdiction
 .ExactMatch = mbolExactMatch
 End With
 Set IClause_Clone = bcClone
End Function

Private Property Set IClause_NextClause(RHS As IClause)
 Set mbcNextClause = RHS
End Property

Private Property Get IClause_NextClause() As IClause
 Set IClause_NextClause = mbcNextClause
End Property

Private Property Set IClause_PriorClause(RHS As IClause)
 Set mbcPriorClause = RHS
End Property

Private Property Get IClause_PriorClause() As IClause
 Set IClause_PriorClause = mbcPriorClause
End Property

Private Property Get IClause_Text() As String
 IClause_Text = Text
End Property

Private Function IClause_Score() As Single

 Dim sngScoreInternal As Single
 Dim sngScoreContext As Single
 Dim strNextWord As String
 Dim strPriorWord As String

 '-- Award up to 1 point for the internal score if the jurisdiction is spelled correctly.

 If mbolExactMatch = True Then

 sngScoreInternal = 0.75

 If ActiveDoc.Words(mlngJurStart, mlngJurEnd) = Jurisdiction.AbrvName Then sngScoreInternal = sngScoreInternal + 0.25

 End If

```

'-- Award up to .5 points for the context score for the expected preceeding clause and text.
If Not (mbcPriorClause Is Nothing) Then
    If Not (mlngJurStart - 1 = mbcPriorClause.ClauseEnd) Then strPriorWord = ActiveDoc.Words(mlngJurStart - 1)
    If TypeOf mbcPriorClause Is cClauseReporter Then sngScoreContext = sngScoreContext + 0.25
    Select Case strPriorWord
        Case "("
            sngScoreContext = sngScoreContext + 0.25
        Case ",", "
            sngScoreContext = sngScoreContext + 0.125
    End Select
End If

'-- Award up to .5 points for the context score for the expected subsequent clause and text.
If Not (mbcNextClause Is Nothing) Then
    If Not (mlngJurEnd + 1 = mbcNextClause.ClauseStart) Then strNextWord = ActiveDoc.Words(mlngJurEnd + 1)
    If TypeOf mbcNextClause Is cClauseCourt Or TypeOf mbcNextClause Is cClauseDate Then
        sngScoreContext = sngScoreContext + 0.25
        Select Case strNextWord
            Case ""
                sngScoreContext = sngScoreContext + 0.25
            Case ",", "
                sngScoreContext = sngScoreContext + 0.125
        End Select
    Else
        Select Case strNextWord
            Case ")"
                sngScoreContext = sngScoreContext + 0.25
            Case " ", " ", "
                sngScoreContext = sngScoreContext + 0.125
        End Select
    End If
Else
    If ActiveDoc.Words(mlngJurEnd + 1) = ")" Then sngScoreContext = sngScoreContext + 0.25
End If

'-- Average the internal and context scores, with 40% weight for the internal score and 60% weight for the context score.
IClause_Score = 0.4 * sngScoreInternal + 0.6 * sngScoreContext

End Function

Friend Function Score() As Single
    Score = IClause_Score()
End Function

Private Sub Class_Initialize()
    ClausesCount = ClausesCount + 1
    Debug.Print "Creating clause. Total: " & ClausesCount
End Sub

Private Sub Class_Terminate()
    ClausesCount = ClausesCount - 1
    Debug.Print "Terminating clause. Total: " & ClausesCount
End Sub

```

```
'-- Copyright 2001 by Robert L. Jacobs
```

```
Option Explicit
```

```
Private mcolClausesAtWord() As Collection
Private mbolPerfectReporter() As Boolean
Private mbolPerfectJurisdiction() As Boolean
Private mbolPerfectCourt() As Boolean
```

```
Private Const MAXIMUMCLAUSELENGTH As Long = 12
```

```
Public Sub Initialize()
```

```
    ReDim mcolClausesAtWord(1 To ActiveDoc.Words.Count)
    ReDim mbolPerfectReporter(1 To ActiveDoc.Words.Count)
    ReDim mbolPerfectJurisdiction(1 To ActiveDoc.Words.Count)
    ReDim mbolPerfectCourt(1 To ActiveDoc.Words.Count)
```

```
End Sub
```

```
Public Function ClausesStartingAtWord(WordNumber As Long) As Collection
```

```
    Dim lngCheckTo As Long
    Dim lngLastWord As Long
    Dim strPhraseTrim As String
    Dim bcNewClause As IClause
```

```
    If mcolClausesAtWord(WordNumber) Is Nothing Then
```

```
        Set mcolClausesAtWord(WordNumber) = New Collection
```

```
        If Not (ActiveDoc.Words.Underline(WordNumber) = False) Then
```

```
            Set bcNewClause = MatchingCaseNameClause(WordNumber)
```

```
            If Not (bcNewClause Is Nothing) Then
```

```
                mcolClausesAtWord(WordNumber).Add bcNewClause
```

```
            End If
```

```
        ElseIf Not (ActiveDoc.Words.WordsTrim(WordNumber) = "") Then
```

```
            '-- Determine the maximum number of words to check in the phrase.
```

```
            lngCheckTo = WordNumber + MAXIMUMCLAUSELENGTH - 1
```

```
            If lngCheckTo > ActiveDoc.Words.Count Then lngCheckTo = ActiveDoc.Words.Count
```

```
            '-- Iterate from the longest phrase to the shortest phrase.
```

```
            For lngLastWord = lngCheckTo To WordNumber Step -1
```

```
                If Not (ActiveDoc.Words.WordsTrim(lngLastWord) = "") Then
```

```
                    strPhraseTrim = ActiveDoc.Words.WordsTrim(WordNumber, lngLastWord)
```

```
                    '-- Locate and add the matching reporter clauses, matching jurisdiction clauses and matching court clauses
```

```
                    '    only if the first and last words are not numbers.
```

```
                    If Not ActiveDoc.Words.IsNumber(WordNumber, False) _
```

```
                    And Not ActiveDoc.Words.IsNumber(lngLastWord, False) Then
```

```
                        Set bcNewClause = MatchingReporterClause(WordNumber, lngLastWord, strPhraseTrim)
```

```
                        If Not (bcNewClause Is Nothing) Then mcolClausesAtWord(WordNumber).Add bcNewClause
```

```
                        Set bcNewClause = MatchingJurisdictionClause(WordNumber, lngLastWord, strPhraseTrim)
```

```
                        If Not (bcNewClause Is Nothing) Then mcolClausesAtWord(WordNumber).Add bcNewClause
```

```
                        Set bcNewClause = MatchingCourtClause(WordNumber, lngLastWord, strPhraseTrim)
```

```
                        If Not (bcNewClause Is Nothing) Then mcolClausesAtWord(WordNumber).Add bcNewClause
```

```
                    End If
```

```
                    '-- Add the matching date clause.
```

```
                    Set bcNewClause = MatchingDateClause(WordNumber, lngLastWord, ActiveDoc.Words(WordNumber, lngLastWord))
```

```
                    If Not (bcNewClause Is Nothing) Then mcolClausesAtWord(WordNumber).Add bcNewClause
```

```
                End If
```

```
            Next lngLastWord
```

```
        Else
```

```
            '-- Add a matching date clause placeholder.
```

```
            Set bcNewClause = MatchingDateClausePlaceholder(WordNumber)
```

```
            If Not (bcNewClause Is Nothing) Then mcolClausesAtWord(WordNumber).Add bcNewClause
```

```
        End If
```

```
    End If
```

```
    Set ClausesStartingAtWord = mcolClausesAtWord(WordNumber)
```

```
End Function
```

```
Private Function MatchingCaseNameClause(FirstWord As Long) As cClauseCaseName
```

```
    Dim i As Long
```

```
    If ActiveDoc.Words(FirstWord, FirstWord + 2) = CASENAME_PLACEHOLDER Then
```

```
        Set MatchingCaseNameClause = New cClauseCaseName
```

```
        MatchingCaseNameClause.ClauseStart = FirstWord
```

```
        MatchingCaseNameClause.ClauseEnd = FirstWord + 2
```

```
    ElseIf ActiveDoc.Words.Underline(FirstWord - 1) = False Then
```

```
        Set MatchingCaseNameClause = New cClauseCaseName
```

```
        MatchingCaseNameClause.ClauseStart = FirstWord
```

```
        For i = FirstWord To ActiveDoc.Words.Count - 1
```

```
            If ActiveDoc.Words.Underline(i + 1) = False Then
```

```
                MatchingCaseNameClause.ClauseEnd = i
```

```
            Exit For
```

```

        End If
    Next i
End If

End Function

Private Function MatchingReporterClause(FirstWord As Long, LastWord As Long, PhraseTrim As String) As cClauseReporter

    Dim bcReporter As cReporter
    Dim bcClause As IClause
    Dim bcExistingReporterClause As cClauseReporter
    Dim bolExactMatch

    '-- Abort if a correctly-spelled reporter has already been found that overlaps this phrase.
    If ExactMatchFound(mbolPerfectReporter(), FirstWord, LastWord) Then Exit Function

    '-- Abort if the phrase is "id", since the phrase is mistaken for "Ind."
    If PhraseTrim = "id" Then Exit Function

    '-- Is there a reporter that matches this phrase?
    Set bcReporter = Jurisdictions.AllReporters.Item(PhraseTrim, True, False, True)
    If Not (bcReporter Is Nothing) Then
        ExactMatchFound(mbolPerfectReporter(), FirstWord, LastWord) = True
        bolExactMatch = True
    Else
        Set bcReporter = Jurisdictions.AllReporters.Item(PhraseTrim, False, True, True)
    End If

    If Not (bcReporter Is Nothing) Then

        '-- Abort if there's an existing reporter clause starting at the same word with the same associated reporter.
        For Each bcClause In mcolClausesAtWord(FirstWord)
            If TypeOf bcClause Is cClauseReporter Then
                Set bcExistingReporterClause = bcClause
                If bcExistingReporterClause.Reporter Is bcReporter Then
                    If bcExistingReporterClause.ClauseEnd = LastWord + 1 Then
                        Exit Function
                    End If
                End If
            End If
        Next bcClause

        '-- Build a new reporter clause.
        Set MatchingReporterClause = New cClauseReporter
        With MatchingReporterClause
            Set .Reporter = bcReporter
            .AbrrvStart = FirstWord
            .AbrrvEnd = LastWord
            If ActiveDoc.Words(LastWord + 1) = "." Then .AbrrvEnd = LastWord + 1
            .ExactMatch = bolExactMatch
        End With
    End If
End Function

Private Function MatchingJurisdictionClause(FirstWord As Long, LastWord As Long, PhraseTrim As String) As cClauseJurisdiction

    Dim bcJurisdiction As cJurisdiction
    Dim bcClause As IClause
    Dim bcExistingJurisdictionClause As cClauseJurisdiction
    Dim bolExactMatch As Boolean

    '-- Abort if a correctly-spelled jurisdiction has already been found that overlaps this phrase.
    If ExactMatchFound(mbolPerfectJurisdiction(), FirstWord, LastWord) Then Exit Function

    '-- Abort if the phrase is "id", since the phrase is mistaken for "Ind."
    If PhraseTrim = "id" Then Exit Function

    '-- Is there a jurisdiction that matches this phrase?
    Set bcJurisdiction = Jurisdictions.Item(PhraseTrim, True, False, True)
    If Not (bcJurisdiction Is Nothing) Then
        ExactMatchFound(mbolPerfectJurisdiction(), FirstWord, LastWord) = True
        bolExactMatch = True
    Else
        Set bcJurisdiction = Jurisdictions.Item(PhraseTrim, False, True, True)
    End If

    If Not (bcJurisdiction Is Nothing) Then

        '-- Abort if there's an existing jurisdiction clause starting at the same word with the same associated reporter.
        For Each bcClause In mcolClausesAtWord(FirstWord)
            If TypeOf bcClause Is cClauseJurisdiction Then
                Set bcExistingJurisdictionClause = bcClause
                If bcExistingJurisdictionClause.Jurisdiction Is bcJurisdiction Then Exit Function
            End If
        Next bcClause

        '-- Build a new reporter clause.
        Set MatchingJurisdictionClause = New cClauseJurisdiction
        With MatchingJurisdictionClause
            Set .Jurisdiction = bcJurisdiction
            .JurStart = FirstWord
            .JurEnd = LastWord
            If ActiveDoc.Words(LastWord + 1) = "." Then .JurEnd = LastWord + 1
            .ExactMatch = bolExactMatch
        End With
    End If
End Function

```

```

End If

End Function

Private Function MatchingCourtClause(FirstWord As Long, LastWord As Long, PhraseTrim As String) As cClauseCourt
    Dim bcCourt As cCourt
    Dim bcClause As cClause
    Dim bcExistingCourtClause As cClauseCourt
    Dim bolExactMatch As Boolean

    '-- Abort if a correctly-spelled jurisdiction has already been found that overlaps this phrase.
    If ExactMatchFound(mbolPerfectCourt(), FirstWord, LastWord) Then Exit Function

    '-- Is there a court that matches this phrase?
    Set bcCourt = Jurisdictions.AllCourts.Item(PhraseTrim, True, False, True)
    If Not (bcCourt Is Nothing) Then
        ExactMatchFound(mbolPerfectCourt(), FirstWord, LastWord) = True
        bolExactMatch = True
    Else
        Set bcCourt = Jurisdictions.AllCourts.Item(PhraseTrim, False, True, True)
    End If

    If Not (bcCourt Is Nothing) Then
        '-- Abort if there's an existing jurisdiction clause starting at the same word with the same associated reporter.
        For Each bcClause In mcolClausesAtWord(FirstWord)
            If TypeOf bcClause Is cClauseCourt Then
                Set bcExistingCourtClause = bcClause
                If bcExistingCourtClause.Court Is bcCourt Then Exit Function
            End If
        Next bcClause

        Set MatchingCourtClause = New cClauseCourt
        With MatchingCourtClause
            Set .Court = bcCourt
            .CourtStart = FirstWord
            .CourtEnd = LastWord
            If ActiveDoc.Words(LastWord + 1) = "." Then .CourtEnd = LastWord + 1
            .ExactMatch = bolExactMatch
        End With
    End If

End Function

Private Function MatchingDateClausePlaceholder(WordNumber As Long) As cClauseDate
    If ActiveDoc.Words(WordNumber) = "____" Then
        Set MatchingDateClausePlaceholder = New cClauseDate
        With MatchingDateClausePlaceholder
            .DateStart = WordNumber
            .DateEnd = WordNumber
        End With
    End If

End Function

Private Function MatchingDateClause(FirstWord As Long, LastWord As Long, ByVal Phrase As String) As cClauseDate
    Dim lngLastWordVal As Long

    '-- IsDate does not interpret a valid date that is missing a space after a period; e.g., "Jan.1, 1980".
    ' Force an extra space after each period to compensate.
    Phrase = Replace(Phrase, ".", ". ")

    lngLastWordVal = Val(ActiveDoc.Words(LastWord))
    If lngLastWordVal > 1600 And lngLastWordVal < 2010 Then
        If LastWord = FirstWord Then
            Set MatchingDateClause = New cClauseDate
            With MatchingDateClause
                .DateStart = FirstWord
                .DateEnd = LastWord
                .Year = lngLastWordVal
            End With
        ElseIf IsDate(Phrase) Then
            Set MatchingDateClause = New cClauseDate
            With MatchingDateClause
                .DateStart = FirstWord
                .DateEnd = LastWord
                .FullDate = CDate(Phrase)
                .Year = CLng(Format(Phrase, "yyyy"))
            End With
        End If
    End If

End Function

Private Property Get ExactMatchFound(PerfectClause() As Boolean, FirstWord As Long, LastWord As Long) As Boolean
    Dim i As Long

    ExactMatchFound = True
    For i = FirstWord To LastWord
        If PerfectClause(i) = False Then
            ExactMatchFound = False
            Exit Property
        End If
    Next i
End Property

```

[illegible]

```
Private Property Let ExactMatchFound(PerfectClause() As Boolean, FirstWord As Long, LastWord As Long, Source As Boolean)
```

```
For i = FirstWord To LastWord
    PerfectClause(i) = Source
Next i
```

[illegible]

cClauseReporter - 1

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit
Implements IClause

Public IsParenthetical As Boolean

Private mlngFirstPageLocation As Long
Private mlngVolumeNumberLocation As Long
Private mlngPinStart As Long
Private mlngPinEnd As Long
Private mbcReporter As cReporter

Private mlngAbrvStart As Long
Private mlngAbrvEnd As Long
Private mbcNextClause As IClause
Private mbcPriorClause As IClause

Private mbcEditor As cEditor

Private mbolExactMatch As Boolean

Friend Property Let ExactMatch(Source As Boolean)
 mbolExactMatch = Source
End Property

Public Property Get FirstPageLocation() As Long
 FirstPageLocation = mlngFirstPageLocation
End Property

Friend Property Let FirstPageLocation(Source As Long)
 mlngFirstPageLocation = Source
End Property

Public Property Get VolumeNumberLocation() As Long
 VolumeNumberLocation = mlngVolumeNumberLocation
End Property

Friend Property Let VolumeNumberLocation(Source As Long)
 mlngVolumeNumberLocation = Source
End Property

Public Property Get PinStart() As Long
 PinStart = mlngPinStart
End Property

Friend Property Let PinStart(Source As Long)
 mlngPinStart = Source
End Property

Public Property Get PinEnd() As Long
 PinEnd = mlngPinEnd
End Property

Friend Property Let PinEnd(Source As Long)
 mlngPinEnd = Source
End Property

Public Property Get Reporter() As cReporter
 Set Reporter = mbcReporter
End Property

Friend Property Set Reporter(Source As cReporter)
 Set mbcReporter = Source
End Property

Public Property Get AbrvStart() As Long
 AbrvStart = mlngAbrvStart
End Property

Friend Property Let AbrvStart(Source As Long)
 mlngAbrvStart = Source
End Property

Public Property Get AbrvEnd() As Long
 AbrvEnd = mlngAbrvEnd
End Property

Friend Property Let AbrvEnd(Source As Long)
 mlngAbrvEnd = Source
End Property

Public Property Get EditorClause() As cEditor
 Set EditorClause = mbcEditor
End Property

Friend Property Set EditorClause(Source As cEditor)
 Set mbcEditor = Source
End Property

Public Property Get Text() As String
 Text = ActiveDoc.Words(ClauseStart, ClauseEnd)
End Property

Public Property Get PriorClause() As IClause
 Set PriorClause = mbcPriorClause
End Property

Public Property Get NextClause() As IClause
 Set NextClause = mbcNextClause

cClauseReporter - 2

End Property

```
Public Function ClauseStart() As Long
    If VolumeNumberLocation > 0 Then
        ClauseStart = VolumeNumberLocation
    Else
        ClauseStart = AbrvStart
    End If
End Function
```

```
Public Function ClauseEnd() As Long
    If PinEnd > 0 Then
        ClauseEnd = PinEnd
    ElseIf FirstPageLocation > 0 Then
        ClauseEnd = FirstPageLocation
    ElseIf Not (mbcEditor Is Nothing) Then
        ClauseEnd = mbcEditor.ClauseEnd
    Else
        ClauseEnd = mlngAbrvEnd
    End If
End Function
```

```
Public Function AbrvText() As String
    AbrvText = ActiveDoc.Words(mlngAbrvStart, mlngAbrvEnd)
End Function
```

```
Public Function VolumeNumber() As Long
    If mlngVolumeNumberLocation > 0 Then VolumeNumber = Val(ActiveDoc.Words(mlngVolumeNumberLocation))
End Function
```

```
Public Function FirstPageNumber() As Long
    If mlngFirstPageLocation > 0 Then FirstPageNumber = Val(ActiveDoc.Words(mlngFirstPageLocation))
End Function
```

```
Private Property Get IClause_ClauseEnd() As Long
    IClause_ClauseEnd = ClauseEnd
End Property
```

```
Private Property Get IClause_ClauseStart() As Long
    IClause_ClauseStart = ClauseStart
End Property
```

```
Private Function IClause_Clone() As IClause
```

```
    Dim bcClone As cClauseReporter
    Set bcClone = New cClauseReporter
    With bcClone
        .AbrvStart = mlngAbrvStart
        .AbrvEnd = mlngAbrvEnd
        Set .Reporter = Reporter
        .ExactMatch = mbolExactMatch
    End With
    Set IClause_Clone = bcClone
End Function
```

```
Private Property Set IClause_NextClause(RHS As IClause)
    Set mbcNextClause = RHS
End Property
```

```
Private Property Get IClause_NextClause() As IClause
    Set IClause_NextClause = mbcNextClause
End Property
```

```
Private Property Set IClause_PriorClause(RHS As IClause)
    Set mbcPriorClause = RHS
End Property
```

```
Private Property Get IClause_PriorClause() As IClause
    Set IClause_PriorClause = mbcPriorClause
End Property
```

```
Private Property Get IClause_Text() As String
    IClause_Text = Text
End Property
```

```
Private Function IClause_Score() As Single
```

```
    Dim sngScoreContext As Single
    Dim sngScoreInternal As Single
    Dim strNextWord As String
```

```
    '-- Award up to .2 point for the internal score if the reporter is spelled correctly.
    If mbolExactMatch = True Then
        sngScoreInternal = 0.1
        If ActiveDoc.Words(Me.AbrvStart, Me.AbrvEnd) = Me.Reporter.AbrvName Then sngScoreInternal = sngScoreInternal + 0.1
    End If
```

```
    '-- Award .4 points for the internal score if the reporter has a volume number.
    If ActiveDoc.Words.IsNumber(Me.AbrvStart - 1, True) Then sngScoreInternal = sngScoreInternal + 0.4
```

```
    '-- Award .4 points for the internal score if the reporter has a first page number or pinpoint page number.
    If FirstPageLocation > 0 Then
        sngScoreInternal = sngScoreInternal + 0.4
    ElseIf PinStart > 0 Then
        If ActiveDoc.Words.Trim(PinStart) = "at" Then
```


cClauseReporter - 3

```

    sngScoreInternal = sngScoreInternal + 0.4
Else
    sngScoreInternal = sngScoreInternal + 0.2
End If
End If

'-- Award up to .5 points for the context score for the expected preceeding clause and text.
If Not (mbcPriorClause Is Nothing) Then
    If TypeOf mbcPriorClause Is cClauseReporter Or TypeOf mbcPriorClause Is cClauseCaseName Then sngScoreContext = sngScoreContext + 0.
25 End If
    If ActiveDoc.Words(ClauseStart() - 1) = "," Then sngScoreContext = sngScoreContext + 0.25

'-- Award up to .5 points for the context score for the expected subsequent clause and text.
If Not (mbcNextClause Is Nothing) Then
    If TypeOf mbcNextClause Is cClauseReporter Or TypeOf mbcNextClause Is cClauseJurisdiction Or TypeOf mbcNextClause Is cClauseCourt Or
r TypeOf mbcNextClause Is cClauseDate Then sngScoreContext = sngScoreContext + 0.25
    End If
    strNextWord = ActiveDoc.Words(ClauseEnd() + 1)
    If strNextWord = "," Or strNextWord = "(" Then sngScoreContext = sngScoreContext + 0.25

'-- Average the internal and context scores, with 60% weight for the internal score and 40% weight for the context score.
'    (Greater weight to the internal score than for other clauses because there is more data -- volume numbers and page
'    numbers -- on which to base the internal score.)

IClause_Score = 0.7 * sngScoreInternal + 0.3 * sngScoreContext

End Function

Friend Function Score() As Single
    Score = IClause_Score()
End Function

Private Sub Class_Initialize()
'
'    ClausesCount = ClausesCount + 1
'    Debug.Print "Creating clause. Total: " & ClausesCount
'
End Sub

Private Sub Class_Terminate()
'
'    ClausesCount = ClausesCount - 1
'    Debug.Print "Terminating clause. Total: " & ClausesCount
'
End Sub

```

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Private mcolClauses As Collection
Private mbcLastClause As IClause

Private Sub Class_Initialize()
Set mcolClauses = New Collection
End Sub

Public Sub Add(Clause As IClause, Optional Key As Variant)

If Not (mbcLastClause Is Nothing) Then
Set mbcLastClause.NextClause = Clause
Set Clause.PriorClause = mbcLastClause
End If
mcolClauses.Add Clause, Key
Set mbcLastClause = Clause

End Sub

Public Function Count() As Long
Count = mcolClauses.Count
End Function

Public Function Item(Index As Variant) As IClause
Set Item = mcolClauses.Item(Index)
End Function

Public Function NewEnum() As IUnknown
Set NewEnum = mcolClauses._NewEnum
End Function

Private Sub Class_Terminate()

Dim bcClause As IClause

For Each bcClause In mcolClauses
Set bcClause.NextClause = Nothing
Next bcClause

End Sub

```
'-- Copyright 2001 by Robert L. Jacobson.
```

```
Option Explicit
```

```
Private mlngAbrvStart As Long
Private mlngAbrvEnd As Long
Private mbcReporter As cReporter
Private mlngVolumeNumberLocation As Long
Private mlngVolumeNumber As Long
Private mlngClauseStart As Long
Private mlngClauseEnd As Long
Private mbolInitialized As Boolean
```

```
Public Property Get AbrvStart() As Long
    AbrvStart = mlngAbrvStart
End Property
```

```
Friend Property Let AbrvStart(Source As Long)
    mlngAbrvStart = Source
    If mlngAbrvStart > 0 And mlngAbrvEnd > 0 And Not mbolInitialized Then Call Initialize
End Property
```

```
Public Property Get AbrvEnd() As Long
    AbrvEnd = mlngAbrvEnd
End Property
```

```
Friend Property Let AbrvEnd(Source As Long)
    mlngAbrvEnd = Source
    If mlngAbrvStart > 0 And mlngAbrvEnd > 0 And Not mbolInitialized Then Call Initialize
End Property
```

```
Public Property Get Reporter() As cReporter
    Set Reporter = mbcReporter
End Property
```

```
Friend Property Set Reporter(Source As cReporter)
    Set mbcReporter = Source
End Property
```

```
Public Property Get VolumeNumberLocation() As Long
    VolumeNumberLocation = mlngVolumeNumberLocation
End Property
```

```
Public Property Get VolumeNumber() As Long
    VolumeNumber = mlngVolumeNumber
End Property
```

```
Public Property Get ClauseStart() As Long
    ClauseStart = mlngClauseStart
End Property
```

```
Public Property Get ClauseEnd() As Long
    ClauseEnd = mlngClauseEnd
End Property
```

```
Public Function Text() As String
    Text = ActiveDoc.Words(mlngClauseStart, mlngClauseEnd)
End Function
```

```
Public Function AbrvText() As String
    AbrvText = ActiveDoc.Words(mlngAbrvStart, mlngAbrvEnd)
End Function
```

```
Private Sub Initialize()
```

```
    '-- Locate the volume number.
    If ActiveDoc.Words.IsNumber(mlngAbrvStart - 1, True) Then
        mlngVolumeNumberLocation = mlngAbrvStart - 1
        mlngVolumeNumber = Val(ActiveDoc.Words(mlngVolumeNumberLocation))
    ElseIf ActiveDoc.Words.Trim(mlngAbrvStart - 1) = "" And ActiveDoc.Words(mlngAbrvStart - 1) <> "(" Then
        If ActiveDoc.Words.IsNumber(mlngAbrvStart - 2, True) Then
            mlngVolumeNumberLocation = mlngAbrvStart - 2
            mlngVolumeNumber = Val(ActiveDoc.Words(mlngVolumeNumberLocation))
        End If
    End If
```

```
    '-- Locate the clause start.
    mlngClauseStart = mlngAbrvStart
    If mlngVolumeNumberLocation > 0 Then mlngClauseStart = mlngVolumeNumberLocation
    If ActiveDoc.Words(mlngClauseStart - 1) = "(" Then
        mlngClauseStart = mlngClauseStart - 1
    ElseIf ActiveDoc.Words.Trim(mlngClauseStart - 1) = "" And ActiveDoc.Words(mlngClauseStart - 1) <> ")" Then
        If ActiveDoc.Words(mlngClauseStart - 2) = "(" Then
            mlngClauseStart = mlngClauseStart - 2
        End If
    End If
```

```
    '-- Locate the clause end.
    mlngClauseEnd = mlngAbrvEnd
    If ActiveDoc.Words(mlngClauseEnd + 1) = ")" Then
        mlngClauseEnd = mlngClauseEnd + 1
    ElseIf ActiveDoc.Words.Trim(mlngClauseEnd + 1) = "" And ActiveDoc.Words(mlngClauseEnd + 1) <> "(" Then
        If ActiveDoc.Words(mlngClauseEnd + 2) = ")" Then
            mlngClauseEnd = mlngClauseEnd + 2
        ElseIf ActiveDoc.Words(mlngClauseEnd + 1) = "," Then
            mlngClauseEnd = mlngClauseEnd + 1
        End If
    End If
```

```
End Sub
```

```
Friend Function EditorScore(MainClauseAbrvEnd As Long) As Long
    Dim strWord As String

    strWord = ActiveDoc.Words(mlngClauseEnd)
    If strWord = ")" Then
        EditorScore = EditorScore + 2
    ElseIf strWord = "," Then
        EditorScore = EditorScore + 1
    End If

    If mlngVolumeNumberLocation > 0 Then
        If mlngVolumeNumberLocation = mlngAbrvStart - 1 Then EditorScore = EditorScore + 2 Else EditorScore = EditorScore + 1
    End If

    If ActiveDoc.Words(mlngClauseStart) = "(" Then
        EditorScore = EditorScore + 2
    ElseIf ActiveDoc.Words(mlngClauseStart - 1) = "," Then
        EditorScore = EditorScore + 1
    End If

    If mbcReporter.IsEditorReporter Then EditorScore = EditorScore + 3

    If MainClauseAbrvEnd < mlngClauseStart - 1 Then EditorScore = EditorScore - (mlngClauseStart - MainClauseAbrvEnd) + 1
End Function
```

1
 2
 3
 4
 5
 6
 7
 8
 9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100

'-- Copyright 2001 by Robert L. Jackson.

Option Explicit

Public Function IsDecisionClauseClause(Clause As IClause) As Boolean

```
If TypeOf Clause Is cClauseJurisdiction _
Or TypeOf Clause Is cClauseCourt _
Or TypeOf Clause Is cClauseDate Then IsDecisionClauseClause = True
```

End Function

Public Function ReportersCompatibleWithCourt(Court As cCourt, ReporterClauses As Collection, Optional MatchingYear As Long) As Long

'-- Returns one point for each reporter clause that is compatible with the supplied court.

```
Dim bcReporterClause As cClauseReporter
Dim bcMatchingReporter As cReporter
Dim lngScore As Long
```

'-- If the reporter clause's abbreviation is spelled correctly, then only search for reporters within the court that match this abbreviation exactly. Otherwise, search for similar reporters.

For Each bcReporterClause In ReporterClauses

```
If ActiveDoc.Words.WordsTrim(bcReporterClause.AbrvStart, bcReporterClause.AbrvEnd) = bcReporterClause.Reporter.AbrvNameTrim Then
Set bcMatchingReporter = Court.Reporters.Item(bcReporterClause.AbrvText, True, False)
Else
Set bcMatchingReporter = Court.Reporters.Item(bcReporterClause.AbrvText, True, True)
End If
```

If Not (bcMatchingReporter Is Nothing) Then

lngScore = lngScore + 1

If MatchingYear > 0 Then If bcMatchingReporter.YearStart <= MatchingYear And MatchingYear <= bcMatchingReporter.YearEnd Then lngScore = lngScore + 1

End If

Next bcReporterClause

ReportersCompatibleWithCourt = lngScore

End Function

Public Function JurisdictionCompatibilityWithReporters(ByRef Jurisdiction As cJurisdiction, ByRef ReporterClauses As Collection) As Long

'Returns a number indicating the level of compatibility between the jurisdiction and the reporter clauses. (One point for each reporter with which the jurisdiction is compatible.)

```
Dim bcReporterClause As cClauseReporter
Dim strReporterTextTrim As String
Dim bcCourtGroup As cCourtGroup
Dim bcMatchingReporter As cReporter
Dim lngScore As Long
```

For Each bcReporterClause In ReporterClauses

strReporterTextTrim = ActiveDoc.Words.WordsTrim(bcReporterClause.AbrvStart, bcReporterClause.AbrvEnd)

For Each bcCourtGroup In Jurisdiction.CourtGroups

Set bcMatchingReporter = bcCourtGroup.Reporters.Item(strReporterTextTrim, True, True)

If Not (bcMatchingReporter Is Nothing) Then

lngScore = lngScore + 1

Exit For

End If

Next bcCourtGroup

Next bcReporterClause

JurisdictionCompatibilityWithReporters = lngScore

End Function

IClause - 1

101

'-- Copyright 2001 by Robert H. Jacobson.

Option Explicit

Public Property Get Text() As String
End Property

Public Property Get ClauseStart() As Long
End Property

Public Property Get ClauseEnd() As Long
End Property

Public Property Get NextClause() As IClause
End Property

Public Property Set NextClause(Source As IClause)
End Property

Public Property Get PriorClause() As IClause
End Property

Public Property Set PriorClause(Source As IClause)
End Property

Public Function Score() As Single
End Function

Public Function Clone() As IClause
End Function

Public Property Get ClauseStart() As Long
End Property
Public Property Get ClauseEnd() As Long
End Property
Public Property Get NextClause() As IClause
End Property
Public Property Set NextClause(Source As IClause)
End Property
Public Property Get PriorClause() As IClause
End Property
Public Property Set PriorClause(Source As IClause)
End Property
Public Function Score() As Single
End Function
Public Function Clone() As IClause
End Function

mPublic - 1

-- Copyright 2001 by Robert L. Jacobson.

102

Option Explicit

Public ClausesCount As Long

Public Const CASENAME_PLACEHOLDER As String = "_____ v. _____"

Public Const CASENAME_PLACEHOLDER As String = "_____ v. _____"

Option Explicit

End ~~Sub~~

[illegible]

cGlobals - 1

'-- Copyright 2001 by Robert L. Jacobson.

104

Option Explicit

Public Property Get ActiveDoc() As cDocument

Set ActiveDoc = gbcActiveDoc

End Property

Public Property Set ActiveDoc(Source As cDocument)

Set gbcActiveDoc = Source

End Property

Public Property Get ActiveDoc() As cDocument
Set ActiveDoc = gbcActiveDoc
End Property
Public Property Set ActiveDoc(Source As cDocument)
Set gbcActiveDoc = Source
End Property

cRichText - 1

'-- Copyright 2001 by Robert L. Johnson.

105

Option Explicit

Private mrtfBox As RichTextBox

Public Property Get TextRTF() As String

```
    With mrtfBox
        .SelStart = 0
        .SelLength = Len(.Text)
        TextRTF = .SelRTF
    End With
```

End Property

Public Property Let TextRTF(Source As String)

```
    mrtfBox.TextRTF = Source
End Property
```

Public Property Get Text() As String

```
    Text = mrtfBox.Text
End Property
```

Public Property Let SelStart(Source As Long)

```
    mrtfBox.SelStart = Source - 1
End Property
```

Public Property Let SelLength(Source As Long)

```
    mrtfBox.SelLength = Source
End Property
```

Public Property Let SelColor(Source As Long)

```
    mrtfBox.SelColor = Source
End Property
```

Public Function StripRTF(Source As String) As String

```
    TextRTF = Source
    StripRTF = TextRTF
```

End Function

Private Sub Class_Initialize()

```
    Set mrtfBox = fRichText.rtfBox
```

End Sub

Private Sub Class_Terminate()

```
    mrtfBox.TextRTF = ""
```

End Sub

```
'-- Copyright 2001 by Robert L. Jacob
```

Option Explicit

```
Private mmswDocument As Word.Document
Private mstrWordCache() As String
Private mstrWordTrimCache() As String
Private mlngFirstCharCache() As Long
Private mlngLastCharCache() As Long
Private mlngScore() As Long
Private mstrText As String
Private mlngCount As Long
Private mbolCacheInitialized As Boolean
```

```
Friend Sub Initialize(MSWDocument As Word.Document)
    Set mmswDocument = MSWDocument
    mbolCacheInitialized = False
End Sub
```

```
Friend Sub Teardown()
    Set mmswDocument = Nothing
End Sub
```

```
Public Property Get Count() As Long
    If Not mbolCacheInitialized Then Call ResetCache
    Count = mlngCount
End Property
```

```
Public Property Get TextRTF(ByVal FirstWord As Long, Optional ByVal LastWord As Long) As String
```

```
    Dim mswRange As Word.Range
    Dim bcRichText As New cRichText
```

```
    If Not mbolCacheInitialized Then Call ResetCache
    If LastWord = 0 Then LastWord = FirstWord
    If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property
```

```
    'Select the text in the word document, then copy it to the clipboard.
```

```
    Clipboard.Clear
    Set mswRange = Range(FirstWord, LastWord, False, False)
    mswRange.Copy
```

```
    'Paste the text into the RTF box, then set the return value to the box's RTF code.
```

```
    With bcRichText
        .TextRTF = Clipboard.GetText(vbCFRTF)
        TextRTF = .TextRTF
    End With
```

```
End Property
```

```
Public Property Let TextRTF(ByVal FirstWord As Long, ByVal LastWord As Long, Source As String)
```

```
    Dim mswRange As Word.Range
    Dim bcRichText As New cRichText
```

```
    If LastWord = 0 Then LastWord = FirstWord
    If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property
```

```
    'Put the source RTF into the RTF box, then copy the contents of the box to the clipboard.
```

```
    Clipboard.Clear
    With bcRichText
        .TextRTF = Source
        Clipboard.SetText .TextRTF, vbCFRTF
    End With
```

```
    ' Paste the clipboard into the document, then reparse.
```

```
    Set mswRange = Range(FirstWord, LastWord, False, False)
    mswRange.Paste
    mbolCacheInitialized = False
```

```
End Property
```

```
Friend Property Get Text() As String
```

```
    If Not mbolCacheInitialized Then Call ResetCache
    Text = mstrText
```

```
End Property
```

```
Public Property Get Words(ByVal FirstWord As Long, Optional ByVal LastWord As Long) As String
```

```
    Dim lngStart As Long
    Dim lngEnd As Long
```

```
    If Not mbolCacheInitialized Then Call ResetCache
```

```
    If LastWord = 0 Or LastWord = FirstWord Then
```

```
        If FirstWord < 1 Or FirstWord > mlngCount Then Exit Property
```

```
        Words = mstrWordCache(FirstWord)
```

```
    Else
```

```

If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

```

```

lngStart = mlngFirstCharCache(FirstWord) + 1
lngEnd = mlngLastCharCache(LastWord) + 1
Words = Mid$(mstrText, lngStart, lngEnd - lngStart)

```

```
End If
```

```
End Property
```

```
Public Property Let Words(ByVal FirstWord As Long, Optional ByVal LastWord As Long, ByVal NewText As String)
```

```
Dim mswRange As Word.Range
```

```

If Not mbolCacheInitialized Then Call ResetCache
If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

```

```
Set mswRange = Range(FirstWord, LastWord, True, True)
```

```
If Not (Trim$(NewText) = "") Then
```

```

NewText = Trim$(NewText)
If ShouldContainSpace(NewText, Words(LastWord + 1)) Then NewText = NewText & " "
If ShouldContainSpace(Words(FirstWord - 1), NewText) = True Then NewText = " " & NewText

```

```
End If
```

```

mswRange = NewText
mbolCacheInitialized = False

```

```
End Property
```

```
Public Property Get FormattedWords(ByVal FirstWord As Long, Optional ByVal LastWord As Long) As Word.Range
```

```
If Not mbolCacheInitialized Then Call ResetCache
```

```

If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property
Set FormattedWords = Range(FirstWord, LastWord).FormattedText

```

```
End Property
```

```
Public Property Let FormattedWords(ByVal FirstWord As Long, Optional ByVal LastWord As Long, _
ByVal FormattedText As Word.Range)
```

```
Dim mswRange As Word.Range
```

```

If Not mbolCacheInitialized Then Call ResetCache
If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property
If ShouldContainSpace(FormattedText, Words(LastWord + 1)) Then FormattedText = FormattedText & " "
If FirstWord > 1 Then
Set mswRange = mmswDocument.Range(mlngLastCharCache(FirstWord - 1) - 1, mlngFirstCharCache(LastWord + 1) - 1)
If ShouldContainSpace(Words(FirstWord - 1), FormattedText) = True Then FormattedText = " " & FormattedText
Else
Set mswRange = mmswDocument.Range(mlngFirstCharCache(FirstWord) - 1, mlngFirstCharCache(LastWord + 1) - 1)
End If

```

```

mswRange.FormattedText = FormattedText
mbolCacheInitialized = False

```

```
End Property
```

```
Public Property Get WordsTrim(ByVal FirstWord As Long, Optional ByVal LastWord As Long) As String
```

```
Dim i As Long
```

```

If Not mbolCacheInitialized Then Call ResetCache
If FirstWord < 1 Or FirstWord > mlngCount Then Exit Property

```

```
If LastWord = 0 Or LastWord = FirstWord Then
```

```
WordsTrim = mstrWordTrimCache(FirstWord)
```

```
Else
```

```

If LastWord < FirstWord Or LastWord > mlngCount Then Exit Property
If mstrWordTrimCache(FirstWord) = "" Then Exit Property

```

```

For i = FirstWord To LastWord
WordsTrim = WordsTrim & mstrWordTrimCache(i)
Next i

```

```
End If
```

```
End Property
```

```
Public Function IsPlaceholder(ByVal WordNumber As Long) As Boolean
```

```

Dim strWord
Dim strFirstChar As String * 1
Dim i As Long

```

```

If Not mbolCacheInitialized Then Call ResetCache
If WordNumber < 1 Or WordNumber > mlngCount Then Exit Function

strWord = mstrWordCache(WordNumber)

If Len(strWord) = 1 Then
    If strWord = "_" Then
        IsPlaceholder = True
        Exit Function
    End If
Else
    strFirstChar = Left$(strWord, 1)

    If strFirstChar = "-" _
    Or strFirstChar = "-" _
    Or strFirstChar = "x" _
    Or strFirstChar = "X" Then
        IsPlaceholder = True
        For i = 2 To Len(strWord)
            If Mid$(strWord, i, 1) <> strFirstChar Then
                IsPlaceholder = False
                Exit Function
            End If
        Next i
    End If
End If

End Function

Public Function IsNumber(ByVal WordNumber As Long, ByVal IncludePlaceholders As Boolean) As Boolean

    If Not mbolCacheInitialized Then Call ResetCache
    If WordNumber < 1 Or WordNumber > mlngCount Then Exit Function

    If IsNumeric(mstrWordCache(WordNumber)) Then
        IsNumber = True
    ElseIf IncludePlaceholders = True Then
        If IsPlaceholder(WordNumber) Then IsNumber = True
    End If

End Function

Public Property Get FirstChar(ByVal WordNumber As Long) As Long

    If Not mbolCacheInitialized Then Call ResetCache
    If WordNumber < 1 Or WordNumber > mlngCount Then Exit Property

    FirstChar = mlngFirstCharCache(WordNumber)

End Property

Public Property Get LastChar(ByVal WordNumber As Long)

    If Not mbolCacheInitialized Then Call ResetCache
    If WordNumber < 1 Or WordNumber > mlngCount Then Exit Property

    LastChar = mlngLastCharCache(WordNumber)

End Property

Public Property Get SpacesAfterWord(WordNumber As Long) As Long

    If Not mbolCacheInitialized Then Call ResetCache
    If WordNumber < 1 Or WordNumber >= mlngCount Then Exit Property

    SpacesAfterWord = mlngFirstCharCache(WordNumber + 1) - mlngLastCharCache(WordNumber)

End Property

Public Property Get PriorFullWord(ByVal WordNumber As Long) As Long

    Dim i As Long

    If Not mbolCacheInitialized Then Call ResetCache
    If WordNumber < 1 Or WordNumber > mlngCount Then Exit Property

    For i = WordNumber - 1 To 1 Step -1
        If mstrWordTrimCache(i) <> "" Then
            PriorFullWord = i
            Exit Property
        ElseIf IsPlaceholder(i) Then
            PriorFullWord = i
            Exit Property
        End If
    Next i

End Property

Public Property Get NextFullWord(ByVal WordNumber As Long) As Long

    Dim i As Long

```

```

If Not mbolCacheInitialized Then Call ResetCache
If WordNumber < 1 Or WordNumber > mlngCount Then Exit Property

For i = WordNumber + 1 To mlngCount
    If mstrWordTrimCache(i) <> "" Then
        NextFullWord = i
        Exit Property
    ElseIf IsPlaceholder(i) Then
        NextFullWord = i
        Exit Property
    End If
Next i

NextFullWord = mlngCount + 1

End Property

Public Property Get Bold(ByVal FirstWord As Long, Optional ByVal LastWord As Long) As Variant

If Not mbolCacheInitialized Then Call ResetCache
If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

Bold = Range(FirstWord, LastWord).Bold

End Property

Public Property Let Bold(ByVal FirstWord As Long, Optional ByVal LastWord As Long, Bold As Variant)

If Not mbolCacheInitialized Then Call ResetCache
If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

Range(FirstWord, LastWord).Bold = Bold

End Property

Public Property Get Highlight(ByVal FirstWord As Long, Optional ByVal LastWord As Long) As Variant

Dim mswFont As Word.Font

If Not mbolCacheInitialized Then Call ResetCache
If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

Set mswFont = Range(FirstWord, LastWord).Font

If mswFont.Color = wdColorRed Then
    Highlight = True
ElseIf mswFont.Color = wdColorAutomatic Then
    Highlight = False
End If

End Property

Public Property Let Highlight(ByVal FirstWord As Long, Optional ByVal LastWord As Long, ByVal Highlight As Variant)

Dim mswFont As Word.Font

If Not mbolCacheInitialized Then Call ResetCache
If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

Set mswFont = Range(FirstWord, LastWord).Font

If Highlight = True Then
    mswFont.Color = wdColorRed
ElseIf Highlight = False Then
    mswFont.Color = wdColorAutomatic
End If

End Property

Public Property Get Italic(ByVal FirstWord As Long, Optional ByVal LastWord As Long) As Variant

If Not mbolCacheInitialized Then Call ResetCache
If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

Italic = Range(FirstWord, LastWord).Italic

End Property

Public Property Let Italic(ByVal FirstWord As Long, Optional ByVal LastWord As Long, ByVal Italic As Variant)

If Not mbolCacheInitialized Then Call ResetCache
If LastWord = 0 Then LastWord = FirstWord
If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

Range(FirstWord, LastWord).Italic = Italic

End Property

Public Property Get Underline(ByVal FirstWord As Long, Optional ByVal LastWord As Long) As Variant

If Not mbolCacheInitialized Then Call ResetCache
If FirstWord < 1 Or FirstWord > mlngCount Then Exit Property
If LastWord > 0 Then If LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

```

```

    Underline = Range(FirstWord, LastWord).Underline
End Property

Public Property Let Underline(ByVal FirstWord As Long, Optional ByVal LastWord As Long, ByVal Underline As Variant)

    If Not mbolCacheInitialized Then Call ResetCache
    If FirstWord < 1 Or FirstWord > mlngCount Then Exit Property
    If LastWord > 0 Then If LastWord < FirstWord Or LastWord > mlngCount Then Exit Property

    Range(FirstWord, LastWord).Underline = Underline
End Property

Public Sub Delete(ByVal FirstWord As Long, Optional ByVal LastWord As Long)

    Dim bolInsertSpace As Boolean
    Dim mswRange As Range

    If Not mbolCacheInitialized Then Call ResetCache
    If FirstWord < 1 Or FirstWord > mlngCount Then Exit Sub
    If LastWord > 0 Then If LastWord < FirstWord Or LastWord > mlngCount Then Exit Sub

    If ShouldContainSpace(Words(FirstWord - 1), Words(LastWord + 1)) = True Then bolInsertSpace = True

    Set mswRange = Range(FirstWord, LastWord, True, True)
    If bolInsertSpace = True Then
        mswRange.Text = " "
    Else
        mswRange.Delete
    End If

    mbolCacheInitialized = False
End Sub

Public Sub Insert(ByVal InsertBeforeWord As Long, ByVal TextToInsert As Variant)

    Dim mswRange As Word.Range

    If Not mbolCacheInitialized Then Call ResetCache
    If InsertBeforeWord < 1 Or InsertBeforeWord > mlngCount + 1 Then Exit Sub

    If InsertBeforeWord = 1 Then
        Set mswRange = mmswDocument.Range(Start:=0, End:=0)
    ElseIf InsertBeforeWord = mlngCount + 1 Then
        Set mswRange = mmswDocument.Range(mlngLastCharCache(InsertBeforeWord - 1) + 1, mlngLastCharCache(InsertBeforeWord - 1) + 1)
        mswRange.Collapse
    Else
        Set mswRange = mmswDocument.Range(mlngLastCharCache(InsertBeforeWord - 1), mlngFirstCharCache(InsertBeforeWord))
    End If

    If Not Trim(TextToInsert) = "" Then
        TextToInsert = Trim(TextToInsert)
        If ShouldContainSpace(Words(InsertBeforeWord - 1), TextToInsert) = True Then TextToInsert = " " & TextToInsert
        If ShouldContainSpace(TextToInsert, Words(InsertBeforeWord)) Then TextToInsert = TextToInsert & " "
    End If

    If TypeOf TextToInsert Is Range Then
        mswRange.FormattedText = TextToInsert
    Else
        mswRange = TextToInsert
    End If

    mbolCacheInitialized = False
End Sub

Public Function Range(ByVal FirstWord As Long, Optional ByVal LastWord As Long, _
Optional ByVal IncludeLeadingSpaces As Boolean = False, Optional ByVal IncludeTrailingSpace As Boolean = False) _
As Word.Range

    Dim lngStartChar As Long
    Dim lngEndChar As Long

    If Not mbolCacheInitialized Then Call ResetCache
    If LastWord = 0 Then LastWord = FirstWord
    If FirstWord < 1 Or FirstWord > mlngCount Or LastWord < FirstWord Or LastWord > mlngCount Then Exit Function

    If IncludeLeadingSpaces = True And FirstWord > 1 Then
        lngStartChar = mlngLastCharCache(FirstWord - 1)
    Else
        lngStartChar = mlngFirstCharCache(FirstWord)
    End If

    If IncludeTrailingSpace = True And LastWord < mlngCount Then
        lngEndChar = mlngFirstCharCache(LastWord + 1)
    Else
        lngEndChar = mlngLastCharCache(LastWord)
    End If

    Set Range = mmswDocument.Range(lngStartChar, lngEndChar)
End Function

Private Function ShouldContainSpace(ByVal FirstPhrase As Variant, ByVal SecondPhrase As Variant) As Boolean

    Dim strNextChar As String * 1

```

```
Dim strLastChar As String * 1
```

```
If Not mbolCacheInitialized Then Call ResetCache
```

```
FirstPhrase = Trim$(FirstPhrase)
```

```
SecondPhrase = Trim$(SecondPhrase)
```

```
If FirstPhrase = "" Or SecondPhrase = "" Then
```

```
ShouldContainSpace = False
```

```
Else
```

```
strLastChar = Right$(FirstPhrase, 1)
```

```
strNextChar = Left$(SecondPhrase, 1)
```

```
If strLastChar <> "(" _
```

```
And strLastChar <> " " _
```

```
And strNextChar <> ")" _
```

```
And strNextChar <> "]" _
```

```
And strNextChar <> "." _
```

```
And strNextChar <> "," _
```

```
And strNextChar <> ":" _
```

```
And strNextChar <> ";" _
```

```
And strNextChar <> "!" Then
```

```
ShouldContainSpace = True
```

```
End If
```

```
End If
```

```
End Function
```

```
Friend Sub ResetCache()
```

```
If mmswDocument.Range <> mstrText Then
```

```
mngCount = mmswDocument.Characters.Count
```

```
ReDim mstrWordCache(1 To mngCount)
```

```
ReDim mstrWordTrimCache(1 To mngCount)
```

```
ReDim mngFirstCharCache(1 To mngCount)
```

```
ReDim mngLastCharCache(1 To mngCount)
```

```
mstrText = mmswDocument.Range
```

```
Call CacheWords
```

```
End If
```

```
mbolCacheInitialized = True
```

```
End Sub
```

```
Private Sub CacheWords()
```

```
Dim i As Long
```

```
Dim lngChar As Long
```

```
Dim lngNextChar As Long
```

```
Dim lngWordNumber As Long
```

```
Dim strChar As String * 1
```

```
Dim strNextChar As String * 1
```

```
Dim bolStopWord As Boolean
```

```
Dim lngCharType As Long
```

```
Dim lngNextCharType As Long
```

```
Dim lngTextLength As Long
```

```
Dim strWord As String
```

```
Const ALPHANUMERIC As Long = 1
```

```
Const STOPPUNCTUATION As Long = 2
```

```
Const CONTINUEPUNCTUATION As Long = 3
```

```
Const SPACE As Long = 4
```

```
lngWordNumber = 1
```

```
lngTextLength = Len(mstrText)
```

```
strNextChar = Mid$(mstrText, 1, 1)
```

```
lngNextChar = Asc(strNextChar)
```

```
Select Case lngNextChar
```

```
Case 48 To 57, 65 To 90, 97 To 122, 192 To 246, 248 To 255 ' Letters, including foreign letters.
```

```
lngNextCharType = ALPHANUMERIC
```

```
Case 45, 95
```

```
' Underscore or hyphen
```

```
lngNextCharType = CONTINUEPUNCTUATION
```

```
Case 32
```

```
lngNextCharType = SPACE
```

```
Case Else
```

```
lngNextCharType = STOPPUNCTUATION
```

```
End Select
```

```
For i = 1 To lngTextLength
```

```
strChar = strNextChar
```

```
lngChar = lngNextChar
```

```
lngCharType = lngNextCharType
```

```
strNextChar = Mid$(mstrText, i + 1, 1)
```

```
lngNextChar = Asc(strNextChar)
```

```
Select Case lngNextChar
```

```
Case 48 To 57, 65 To 90, 97 To 122, 192 To 246, 248 To 255 ' Letters, including foreign letters.
```

```
lngNextCharType = ALPHANUMERIC
```

```
Case 45, 95
```

```
' Underscore or hyphen
```

```
lngNextCharType = CONTINUEPUNCTUATION
```

```
Case 32
```

```
lngNextCharType = SPACE
```

```
Case Else
```

```
lngNextCharType = STOPPUNCTUATION
```

```
End Select
```



```

Select Case lngCharType
Case ALPHANUMERIC
    strWord = strWord & strChar
    '-- If this character is alphanumeric, then continue the word if the next character is also alphanumeric.
    If lngNextCharType = ALPHANUMERIC Then
        bolStopWord = False
    Else
        bolStopWord = True
    End If
Case CONTINUEPUNCTUATION
    strWord = strWord & strChar
    '-- If this character is a "continue punctuation" (a hyphen or an underscore), then continue the word if the next
    ' character is identical to this character.
    If lngNextChar = lngChar Then
        bolStopWord = False
    Else
        bolStopWord = True
    End If
Case STOPPUNCTUATION
    '-- If this character is a "stop punctuation" (any other type of punctuation), then stop the word.
    strWord = strWord & strChar
    bolStopWord = True
    '-- If this character is a space, do nothing.
End Select

```

```

If bolStopWord = True Or i = lngTextLength Then

```

```

    mstrWordCache(lngWordNumber) = strWord
    If lngCharType = ALPHANUMERIC Then mstrWordTrimCache(lngWordNumber) = LCase$(strWord)
    mlngLastCharCache(lngWordNumber) = i
    mlngFirstCharCache(lngWordNumber) = i - Len(strWord)
    lngWordNumber = lngWordNumber + 1
    bolStopWord = False
    strWord = ""

```

```

End If

```

```

Next i

```

```

mlngCount = lngWordNumber - 1
ReDim Preserve mstrWordCache(1 To mlngCount)
ReDim Preserve mstrWordTrimCache(1 To mlngCount)
ReDim Preserve mlngFirstCharCache(1 To mlngCount)
ReDim Preserve mlngLastCharCache(1 To mlngCount)

```

```

End Sub

```

```

Public Sub ScoreReset()
    ReDim mlngScore(1 To mlngCount)
End Sub

```

```

Public Property Get Score(WordNumber As Long) As Long
    If WordNumber > 0 And WordNumber <= mlngCount Then Score = mlngScore(WordNumber)
End Property

```

```

Public Property Let Score(WordNumber As Long, Source As Long)
    If WordNumber > 0 And WordNumber <= mlngCount Then mlngScore(WordNumber) = Source
End Property

```

```

Public Function ScoreBest(Optional MinimumScore As Long) As Long

```

```

    Dim i As Long
    Dim lngHighestScore As Long

```

```

    lngHighestScore = MinimumScore - 1
    For i = 1 To mlngCount
        If mlngScore(i) > lngHighestScore Then
            lngHighestScore = mlngScore(i)
            ScoreBest = i
        End If
    Next i

```

```

End Function

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2

'-- Copyright 2001 by Robert L. Jacobson.

Option Explicit

Public gbcActiveDoc As cDocument
Public ClausesCount As Long

Public gbcActiveDoc As cDocument
Public ClausesCount As Long

'-- Copyright 2001 by Robert L. Jacob

Option Explicit

Private Const SPACE As Long = 32

Public Function SuperTrim(ByVal Source As String) As String

Dim i As Long

Dim strCharacter As String * 1

Dim strTrimmedString As String

Dim lngAscCharacter As Long

Source = LCase(Source)

For i = 1 To Len(Source)

strCharacter = Mid\$(Source, i, 1)

lngAscCharacter = Asc(strCharacter)

```
If (lngAscCharacter >= 48 And lngAscCharacter <= 57) _
Or (lngAscCharacter >= 97 And lngAscCharacter <= 122) _
Or (lngAscCharacter >= 192 And lngAscCharacter <= 246) _
Or (lngAscCharacter >= 248 And lngAscCharacter <= 255) _
Then strTrimmedString = strTrimmedString & strCharacter
```

```
If lngAscCharacter = 38 Then strTrimmedString = strTrimmedString & "and"
```

Next i

SuperTrim = strTrimmedString

End Function

Public Function ExistsInCollection(Item As Variant, Collection As Collection) As Boolean

Dim varItem As Variant

For Each varItem In Collection

If Item = varItem Then

ExistsInCollection = True

Exit Function

End If

Next varItem

End Function

Public Function AddArticle(Source As String) As String

Returns a string containing the original string, preceeded with its correct indefinite article ("a" or "an").

Dim strChar As String

strChar = LCase(Left\$(Source, 1))

Select Case strChar

Case "a", "e", "i", "o", "u"

AddArticle = "an " & Source

Case Else

AddArticle = "a " & Source

End Select

End Function

```

cTimer - 1

'-- Copyright 2001 by Robert L. Jacob

Option Explicit

Private Declare Function timeGetTime Lib "winmm.dll" () As Long

Private mlngStartTimer As Long
Private mstrIdentifier As String
Private mlngRepetitions As Long

Public Sub StartTimer(Repetitions As Long, Optional ByVal Identifier As String)

    mlngRepetitions = Repetitions
    mstrIdentifier = Identifier
    Debug.Print vbCrLf & "Starting timer...."
    mlngStartTimer = timeGetTime

End Sub

Public Sub EndTimer()

    Dim sngEndTimer As Long
    Dim sngSeconds As Single
    Dim sngLengthPerRep As Single
    Dim dbRepsPerSec As Single

    sngEndTimer = timeGetTime
    sngSeconds = (sngEndTimer - mlngStartTimer) / 1000
    sngLengthPerRep = sngSeconds / mlngRepetitions

    Debug.Print "Ending timer...." & vbCrLf
    If mstrIdentifier <> "" Then Debug.Print "Identifier: " & mstrIdentifier
    Debug.Print "Elapsed seconds: " & sngSeconds

    If sngLengthPerRep > 0 Then
        dbRepsPerSec = Round(mlngRepetitions / sngSeconds, 1)

        If sngLengthPerRep > 1 Then
            sngLengthPerRep = Round(sngLengthPerRep, 2)
            Debug.Print "Length per repetition: " & sngLengthPerRep & " seconds"
            Debug.Print "Repetitions per second: " & dbRepsPerSec
            MsgBox "Repetitions: " & mlngRepetitions & vbCrLf & vbCrLf & "Elapsed seconds: " & sngSeconds & vbCrLf & vbCrLf & "Length per r
            epetition: " & sngLengthPerRep & " seconds" & vbCrLf & vbCrLf & "Repetitions per second: " & dbRepsPerSec, vbInformation, "Timer result."
        ElseIf sngLengthPerRep > 0.001 Then
            sngLengthPerRep = Round(sngLengthPerRep, 4) * 1000
            Debug.Print "Length per repetition: " & sngLengthPerRep & " milliseconds."
            Debug.Print "Repetitions per second: " & dbRepsPerSec
            MsgBox "Repetitions: " & mlngRepetitions & vbCrLf & vbCrLf & "Elapsed seconds: " & sngSeconds & vbCrLf & vbCrLf & "Length per r
            epetition: " & sngLengthPerRep & " milliseconds" & vbCrLf & vbCrLf & "Repetitions per second: " & dbRepsPerSec, vbInformation, "Timer result
            t."
        ElseIf sngSeconds <> 0 Then
            sngLengthPerRep = Round(sngLengthPerRep, 7) * 1000000
            Debug.Print "Length per repetition: " & sngLengthPerRep & " microseconds."
            Debug.Print "Repetitions per second: " & dbRepsPerSec
            MsgBox "Repetitions: " & mlngRepetitions & vbCrLf & vbCrLf & "Elapsed seconds: " & sngSeconds & vbCrLf & vbCrLf & "Length per r
            epetition: " & sngLengthPerRep & " microseconds" & vbCrLf & vbCrLf & "Repetitions per second: " & dbRepsPerSec, vbInformation, "Timer resul
            t."
        End If
    Else
        Debug.Print "Length per repetition: N/A."
        Debug.Print "Repetitions per second: N/A."
        MsgBox "Repetitions: " & mlngRepetitions & vbCrLf & vbCrLf & "Elapsed seconds: Less than one" & vbCrLf & vbCrLf & "Length per repet
            ition: N/A" & vbCrLf & vbCrLf & "Repetitions per second: N/A", vbInformation, "Timer result."
    End If

End Sub

```

cCourt - 1

'-- Copyright 2001 by Robert L. Jacob

Option Explicit

```

Private mstrFullName As String
Private mstrAbrvName As String
Private mstrFullNameTrim As String
Private mstrAbrvNameTrim As String
Private mlngYearStart As Long
Private mlngYearEnd As Long
Private mbcParent As cJurisdiction
Private mbolJurisdictionImplicit As Boolean
Private mbcCourtGroup As cCourtGroup

```

```

Public Score As Long

```

```

'Private Sub Class_Initialize()
'
'    ObjectsCount = ObjectsCount + 1
'    Debug.Print "Court created. Total: " & ObjectsCount
'
'End Sub
'
'Private Sub Class_Terminate()
'
'    ObjectsCount = ObjectsCount - 1
'    Debug.Print "Court destroyed. Total: " & ObjectsCount
'
'End Sub

```

```

Public Property Get FullName() As String
    FullName = mstrFullName
End Property

```

```

Friend Property Let FullName(ByVal Source As String)
    mstrFullName = Source
    mstrFullNameTrim = SuperTrim(Source)
End Property

```

```

Public Property Get AbrvName() As String
    AbrvName = mstrAbrvName
End Property

```

```

Friend Property Let AbrvName(ByVal Source As String)
    mstrAbrvName = Source
    mstrAbrvNameTrim = SuperTrim(Source)
End Property

```

```

Public Property Get FullNameTrim() As String
    FullNameTrim = mstrFullNameTrim
End Property

```

```

Public Property Get AbrvNameTrim() As String
    AbrvNameTrim = mstrAbrvNameTrim
End Property

```

```

Public Property Get YearStart() As Long
    YearStart = mlngYearStart
End Property

```

```

Friend Property Let YearStart(ByVal Source As Long)
    mlngYearStart = Source
End Property

```

```

Public Property Get YearEnd() As Long
    YearEnd = mlngYearEnd
End Property

```

```

Friend Property Let YearEnd(ByVal Source As Long)
    mlngYearEnd = Source
End Property

```

```

Public Property Get Parent() As cJurisdiction
    Set Parent = mbcParent
End Property

```

```

Friend Property Set Parent(Source As cJurisdiction)
    Set mbcParent = Source
End Property

```

```

Public Property Get JurisdictionImplicit() As Boolean
    JurisdictionImplicit = mbolJurisdictionImplicit
End Property

```

```

Friend Property Let JurisdictionImplicit(ByVal Source As Boolean)
    mbolJurisdictionImplicit = Source
End Property

```

```

Public Function ImplicitJurisdiction() As cJurisdiction
    If mbolJurisdictionImplicit Then
        Set ImplicitJurisdiction = Me.Parent
    End If
End Function

```

```

Friend Property Set CourtGroup(Source As cCourtGroup)
    Set mbcCourtGroup = Source
End Property

```

```

Public Property Get CourtGroup() As cCourtGroup

```

```
Public Function NonParallelReporters() As Collection
    Set NonParallelReporters = mbcCourtGroup.NonParallelReporters
End Function
```

and with me from it only if it
 is the best. No. I feel that
 the only way to get the
 best of the world is to get
 the best of the world.

cCourtGroup - 1

'-- Copyright 2001 by Robert L. Jacob

119

Option Explicit

```
Private mbcReporters As cReporters
Private mcolParallelReporters As Collection
Private mcolNonParallelReporters As Collection
Private mbcParent As cJurisdiction
```

Private mstrFullName As String

Private Sub Class_Initialize()

```
Set mbcReporters = New cReporters
Set mcolParallelReporters = New Collection
Set mcolNonParallelReporters = New Collection
```

```
' ObjectsCount = ObjectsCount + 1
' Debug.Print "Court Group created. Total: " & ObjectsCount
```

End Sub

Private Sub Class_Terminate()

```
' ObjectsCount = ObjectsCount - 1
' Debug.Print "Court Group destroyed. Total: " & ObjectsCount
```

End Sub

Public Property Get Reporters() As cReporters

```
Set Reporters = mbcReporters
```

End Property

Friend Property Set Reporters(Source As cReporters)

```
Set mbcReporters = Source
```

End Property

Public Property Get ParallelReporters() As Collection

```
Set ParallelReporters = mcolParallelReporters
```

End Property

Friend Property Set ParallelReporters(Source As Collection)

```
Set mcolParallelReporters = Source
```

End Property

Public Property Get NonParallelReporters() As Collection

```
Set NonParallelReporters = mcolNonParallelReporters
```

End Property

Friend Property Set NonParallelReporters(Source As Collection)

```
Set mcolNonParallelReporters = Source
```

End Property

Public Property Get Parent() As cJurisdiction

```
Set Parent = mbcParent
```

End Property

Friend Property Set Parent(Source As cJurisdiction)

```
Set mbcParent = Source
```

End Property

Public Property Get FullName() As String

```
FullName = mstrFullName
```

End Property

Friend Property Let FullName(ByVal Source As String)

```
mstrFullName = Source
```

End Property

cCourts - 1

'-- Copyright 2001 by Robert L. Jackson.

120

Option Explicit

Private mcolCourts As Collection
Private mbcFuzzyCollection As cFuzzyCollection

Friend Property Get FuzzyCollection() As cFuzzyCollection
Set FuzzyCollection = mbcFuzzyCollection
End Property

Private Sub Class_Initialize()
Set mcolCourts = New Collection
Set mbcFuzzyCollection = New cFuzzyCollection

End Sub

Public Sub ScoreReset()
Dim bcCourt As cCourt
For Each bcCourt In mcolCourts
bcCourt.Score = 0
Next bcCourt

End Sub

Public Function ScoreBest() As cCourt
Dim bcCourt As cCourt
Dim lngBestScore As Long
For Each bcCourt In mcolCourts
If bcCourt.Score > lngBestScore Then
Set ScoreBest = bcCourt
lngBestScore = bcCourt.Score
End If
Next bcCourt
End Function

Public Function ScoresBest(Optional MinimumScore As Long, Optional Deviance As Long) As Collection
Dim bcCourt As cCourt
Dim lngBestScore As Long
Dim lngMinimum As Long
For Each bcCourt In mcolCourts
If bcCourt.Score > lngBestScore Then lngBestScore = bcCourt.Score
Next bcCourt
lngMinimum = lngBestScore - Deviance
If MinimumScore > 0 And lngMinimum < MinimumScore Then lngMinimum = MinimumScore
Set ScoresBest = New Collection
If lngBestScore >= lngMinimum Then
For Each bcCourt In mcolCourts
If bcCourt.Score >= lngMinimum Then ScoresBest.Add bcCourt
Next bcCourt
End If
End Function

Friend Sub Add(Court As cCourt, Aliases As Variant)

Dim varAlias As Variant
Dim strAliasTrim As String

With Court
mcolCourts.Add Court
mbcFuzzyCollection.Add Court, .AbrvNameTrim
If .AbrvNameTrim <> .FullNameTrim Then mbcFuzzyCollection.Add Court, .FullNameTrim
End With

For Each varAlias In Aliases
strAliasTrim = CStr(varAlias)
If Not (varAlias = SuperTrim(strAliasTrim)) Then Stop
mbcFuzzyCollection.Add Court, strAliasTrim
Next varAlias

'-- Aliases should be supplied as trimmed.
'-- For debugging only. Delete later.

End Sub

Public Function Item(Key As String, MatchExactAlias As Boolean, MatchSimilarAlias As Boolean, Optional YearToMatch As Long, Optional KeyIsTrimmed As Boolean = False) As cCourt

Dim strKeyTrim As String
Dim bcCourt As cCourt
Dim colMatches As Collection

If KeyIsTrimmed Then
strKeyTrim = Key
Else
strKeyTrim = SuperTrim(Key)
End If

If MatchExactAlias = False And MatchSimilarAlias = False Then
Set colMatches = New Collection

```

For Each bcCourt In mcolCourts
    If strKeyTrim = bcCourt.Name Then colMatches.Add bcCourt
Next bcCourt
Else
    Set colMatches = mbcFuzzyCollection.ExactItems(strKeyTrim)
    If MatchSimilarAlias = True Then If colMatches.Count = 0 Then Set colMatches = mbcFuzzyCollection.SimilarItems(strKeyTrim)
End If

If colMatches.Count > 0 Then
    '-- Assume that the first match is correct.
    Set Item = colMatches(1)
    '-- If there are multiple matches and a year to match is supplied, then search for a matching court with the
    ' correct year.
    If colMatches.Count > 1 And YearToMatch > 0 Then
        For Each bcCourt In colMatches
            If bcCourt.YearStart <= YearToMatch And bcCourt.YearEnd >= YearToMatch Then
                Set Item = bcCourt
                Exit For
            End If
        Next bcCourt
    End If
End If

End Function

Public Function Items(Key As String, MatchSimilarFuzzyCollection As Boolean, Optional KeyIsTrimmed As Boolean = False) As Collection

    Dim strKeyTrim As String
    Dim bcCourt As cCourt

    If KeyIsTrimmed Then
        strKeyTrim = Key
    Else
        strKeyTrim = SuperTrim(Key)
    End If

    Set Items = mbcFuzzyCollection.ExactItems(strKeyTrim)
    If MatchSimilarFuzzyCollection = True Then
        If Items.Count = 0 Then Set Items = mbcFuzzyCollection.SimilarItems(strKeyTrim)
    End If

End Function

Public Function Count() As Long
    Count = mcolCourts.Count
End Function

Public Function NewEnum() As IUnknown
    Set NewEnum = mcolCourts.[_NewEnum]
End Function

```

```
-- Copyright 2001 by Robert L. Jacob
```

Option Explicit

```
Private mobjItems() As Object
Private mstrKeys() As String
Private mintArrayStart() As Integer
Private mintArrayEnd() As Integer
Private mlngCount As Long
Private mlngArraySize As Long
Private mbolInitialized As Boolean
```

```
Private Const INCREASE_PERCENT As Long = 0.1
Private Const INCREASE_MIN As Long = 10
Private Const BUCKETS As Long = 128
```

```
Friend Sub Add(Item As Object, Key As String)
```

```
    Dim lngIncrease As Long
```

```
    '-- If the array is full, then resize the array to make room for the new entry.
```

```
    If mlngCount = mlngArraySize Then
        lngIncrease = mlngArraySize * INCREASE_PERCENT
        If lngIncrease < INCREASE_MIN Then lngIncrease = INCREASE_MIN
        mlngArraySize = mlngArraySize + lngIncrease
        ReDim Preserve mobjItems(1 To mlngArraySize)
        ReDim Preserve mstrKeys(1 To mlngArraySize)
    End If
```

```
    '-- Add the new item.
    mlngCount = mlngCount + 1
    Set mobjItems(mlngCount) = Item
    mstrKeys(mlngCount) = Key
```

```
    mbolInitialized = False
```

```
End Sub
```

```
Friend Function ExactItem(Source As String) As Object
```

```
    '-- Look for an item that has a key that matches exactly.
```

```
    Dim i As Long
    Dim lngValue As Long
```

```
    If Not mbolInitialized Then Call Initialize
    lngValue = AsciiValue(Source)
```

```
    For i = mintArrayStart(lngValue) To mintArrayEnd(lngValue)
        If Source = mstrKeys(i) Then
            Set ExactItem = mobjItems(i)
            Exit Function
        End If
    Next i
```

```
End Function
```

```
Friend Function ExactItems(Source As String) As Collection
```

```
    '-- Look for an item that has a key that matches exactly.
```

```
    Dim i As Long
    Dim lngValue As Long
```

```
    Set ExactItems = New Collection
    If Not mbolInitialized Then Call Initialize
    lngValue = AsciiValue(Source)
```

```
    For i = mintArrayStart(lngValue) To mintArrayEnd(lngValue)
        If Source = mstrKeys(i) Then
            ExactItems.Add mobjItems(i)
        End If
    Next i
```

```
End Function
```

```
Friend Function SimilarItems(Source As String) As Collection
```

```
    '-- Search for an item with a similar key.
```

```
    Dim i As Long
    Dim lngValue As Long
    Dim lngSourceLength As Long
```

```
    Set SimilarItems = New Collection
    If Not mbolInitialized Then Call Initialize
    lngSourceLength = Len(Source)
```

```
    If lngSourceLength > 1 Then
        lngValue = AsciiValue(Source)
        For i = mintArrayStart(lngValue) To mintArrayEnd(lngValue)
            Select Case Len(mstrKeys(i))
                Case Is > 2
                    If Abs(lngSourceLength - Len(mstrKeys(i))) < 2 Then
                        If EditDistanceMaxOne(mstrKeys(i), Source) = 1 Then
```

```

        SimilarItems.Add mobjItems(i)
    End If
End If

Case Is = 2
    If lngSourceLength = 3 Then
        If EditDistanceMaxOne(mstrKeys(i), Source) = 1 Then
            SimilarItems.Add mobjItems(i)
        End If
    End If

End Select

Next i
End If

End Function

Friend Sub Initialize()

    Call SortValues
    Call CreateForwardStar
    mbolInitialized = True

End Sub

Private Sub SortValues()

    '-- Sort the items by key value using a modified countingsort.

    Dim i As Long
    Dim lngValue As Long
    Dim lngCounts() As Long
    Dim lngNextPosition() As Long
    Dim lngPosition As Long
    Dim strSortedKeys() As String
    Dim objSortedItems() As Object

    '-- Initialize the counts to 0.
    ReDim lngCounts(0 To BUCKETS)
    ReDim lngNextPosition(0 To BUCKETS)

    '-- Count the values.
    For i = 1 To mlngCount
        lngValue = AsciiValue(mstrKeys(i))
        lngCounts(lngValue) = lngCounts(lngValue) + 1
    Next i

    '-- Determine the correct starting position for each value.
    lngPosition = 1
    For i = 0 To BUCKETS
        lngNextPosition(i) = lngPosition
        lngPosition = lngPosition + lngCounts(i)
    Next i

    '-- Build a new array with the items in their correct positions.
    ReDim strSortedKeys(1 To mlngCount)
    ReDim objSortedItems(1 To mlngCount)
    For i = 1 To mlngCount
        lngValue = AsciiValue(mstrKeys(i))
        lngPosition = lngNextPosition(lngValue)
        If Not strSortedKeys(lngPosition) = vbNullString Then Err.Raise 1000 '-- Check for error.
        strSortedKeys(lngPosition) = mstrKeys(i)
        Set objSortedItems(lngPosition) = mobjItems(i)
        lngNextPosition(lngValue) = lngNextPosition(lngValue) + 1
    Next i

    '-- Assign the new sorted arrays as the main module arrays.
    mstrKeys = strSortedKeys
    mobjItems = objSortedItems
    mlngArraySize = mlngCount

End Sub

Private Sub CreateForwardStar()

    '-- Build a forward star data structure by locating the first and last items containing this first letter

    Dim i As Long
    Dim lngValue As Long

    ReDim mintArrayStart(0 To BUCKETS)
    ReDim mintArrayEnd(0 To BUCKETS)

    For i = 0 To BUCKETS
        mintArrayStart(i) = 0
        mintArrayEnd(i) = -1
    Next i

    For i = 1 To mlngCount
        lngValue = AsciiValue(mstrKeys(i))
        If mintArrayStart(lngValue) = 0 Then mintArrayStart(lngValue) = i
        mintArrayEnd(lngValue) = i
    Next i

End Sub

Private Function AsciiValue(Source As String) As Long

    '-- Returns a value depending on the first letter of the source, or 0 if the source is an empty string.

```

```

    If Source = vbNullString Then
        AsciiValue = 0
    Else
        AsciiValue = Asc(Source) Mod BUCKETS
    End If
End Function

Private Sub DebugPrintForwardStar()

    Dim i As Long

    For i = 0 To BUCKETS
        Debug.Print i & ": " & mintArrayStart(i), mintArrayEnd(i)
    Next i

End Sub

Private Sub DebugPrintItems(Value As Long)

    Dim i As Long

    For i = mintArrayStart(Value) To mintArrayEnd(Value)
        Debug.Print mstrKeys(i)
    Next i

End Sub

Private Function EditDistanceMaxOne(ByVal Source As String, ByVal Target As String) As Long

    Dim i As Long
    Dim lngFirstWrongChar As Long
    Dim lngLastWrongChar As Long
    Dim bytSource() As Byte
    Dim bytTarget() As Byte
    Dim lngSourceLen As Long
    Dim lngTargetLen As Long
    Dim lngOffset As Long

    lngSourceLen = Len(Source)
    lngTargetLen = Len(Target)

    ' Convert the strings to byte arrays. If one string is longer than the other, then pad the end of the shorter string with
    ' an extra space.
    If lngSourceLen < lngTargetLen Then
        bytSource = Source & " "
        bytTarget = Target
        lngOffset = -2
    ElseIf lngSourceLen > lngTargetLen Then
        bytSource = Source
        bytTarget = Target & " "
        lngOffset = 2
    Else
        bytSource = Source
        bytTarget = Target
    End If

    ' Search forward from the start to determine the first error.
    For i = 0 To UBound(bytSource) Step 2
        If bytSource(i) <> bytTarget(i) Then
            lngFirstWrongChar = i
            Exit For
        End If
    Next i

    '-- Search backwards from the end to determine the last error.
    If lngSourceLen > lngTargetLen Then
        For i = lngSourceLen * 2 - 2 To lngFirstWrongChar Step -2
            If bytSource(i) <> bytTarget(i - lngOffset) Then
                lngLastWrongChar = i
                Exit For
            End If
        Next i
    ElseIf lngSourceLen = lngTargetLen Then
        For i = lngSourceLen * 2 - 2 To lngFirstWrongChar Step -2
            If bytSource(i) <> bytTarget(i) Then
                lngLastWrongChar = i
                Exit For
            End If
        Next i
    Else
        For i = lngSourceLen * 2 - 2 To lngFirstWrongChar Step -2
            If bytSource(i) <> bytTarget(i - lngOffset) Then
                lngLastWrongChar = i - lngOffset
                Exit For
            End If
        Next i
    End If

    '-- Determine whether the first and last errors are at the same position.
    If lngFirstWrongChar >= lngLastWrongChar Then
        EditDistanceMaxOne = 1
    ElseIf lngFirstWrongChar = lngLastWrongChar - 2 Then
        '-- If the only problem is that two characters are transposed, treat that as a single edit count
        ' (even though the traditional edit distance function treats transposed characters as two edits).
        If bytSource(lngFirstWrongChar) = bytTarget(lngLastWrongChar) _
            And bytSource(lngLastWrongChar) = bytTarget(lngFirstWrongChar) Then
            EditDistanceMaxOne = 1
        Else

```

cFuzzyCollection - 4

```
        EditDistanceMaxOne = 2
    End If
Else
    EditDistanceMaxOne = 2
End If
```

End Function

125

125

-- Copyright 2001 by Robert L. Jacob

Option Explicit

Private mbcJurisdictions As cJurisdictions

Public Property Get Jurisdictions() As cJurisdictions

If mbcJurisdictions Is Nothing Then Set mbcJurisdictions = New cJurisdictions

Set Jurisdictions = mbcJurisdictions

End Property

Public Property Set Jurisdictions(Source As cJurisdiction)

Set mbcJurisdictions = Source

End Property

Private Sub Class_Initialize()

Set mbcJurisdictions = gbcJurisdictions

End Sub

Public Property Get Jurisdictions() As cJurisdictions
If mbcJurisdictions Is Nothing Then Set mbcJurisdictions = New cJurisdictions
Set Jurisdictions = mbcJurisdictions
End Property
Public Property Set Jurisdictions(Source As cJurisdiction)
Set mbcJurisdictions = Source
End Property
Private Sub Class_Initialize()
Set mbcJurisdictions = gbcJurisdictions
End Sub

Option Explicit

```

Private mbcCourts As cCourts
Private mcolCourtGroups As Collection
Private mbcAllReporters As cReporters
Private mstrFullName As String
Private mstrAbrvName As String
Private mstrFullNameTrim As String
Private mstrAbrvNameTrim As String
Public Score As Long

```

Private Sub Class_Initialize()

```

    Set mbcCourts = New cCourts
    Set mcolCourtGroups = New Collection
    Set mbcAllReporters = New cReporters

    ObjectsCount = ObjectsCount + 1
    Debug.Print "Jurisdiction created. Total: " & ObjectsCount

```

End Sub

'Private Sub Class_Terminate()

```

    ObjectsCount = ObjectsCount - 1
    Debug.Print "Jurisdiction destroyed. Total: " & ObjectsCount

```

'End Sub

Public Property Get Courts() As cCourts

```

    Set Courts = mbcCourts

```

End Property

Public Property Get AllReporters() As cReporters

```

    Set AllReporters = mbcAllReporters

```

End Property

Public Property Get FullName() As String

```

    FullName = mstrFullName

```

End Property

Friend Property Let FullName(ByVal Source As String)

```

    mstrFullName = Source
    mstrFullNameTrim = SuperTrim(Source)

```

End Property

Public Property Get FullNameTrim() As String

```

    FullNameTrim = mstrFullNameTrim

```

End Property

Public Property Get AbrvName() As String

```

    AbrvName = mstrAbrvName

```

End Property

Friend Property Let AbrvName(ByVal Source As String)

```

    mstrAbrvName = Source
    mstrAbrvNameTrim = SuperTrim(Source)

```

End Property

Public Property Get AbrvNameTrim() As String

```

    AbrvNameTrim = mstrAbrvNameTrim

```

End Property

Public Property Get CourtGroups() As Collection

```

    Set CourtGroups = mcolCourtGroups

```

End Property

Option Explicit

```
Private mcolJurisdictions As Collection
Private mcolAllCourtGroups As Collection
Private mbcAllCourts As cCourts
Private mbcAllReporters As cReporters
Private mbcFuzzyCollection As cFuzzyCollection
```

Private Sub Class_Initialize()

```
Set mcolJurisdictions = New Collection
Set mcolAllCourtGroups = New Collection
Set mbcAllCourts = New cCourts
Set mbcAllReporters = New cReporters
Set mbcFuzzyCollection = New cFuzzyCollection
```

```
Call PopulateJurisdictions(Me)
Call LoadReporters(Me)
```

```
Debug.Print "Jurisdictions Initialize"
```

End Sub

Private Sub Class_Terminate()

```
Debug.Print "Jurisdictions Terminate"
```

End Sub

Public Sub ScoreReset()

```
Dim bcJurisdiction As cJurisdiction
For Each bcJurisdiction In mcolJurisdictions
    bcJurisdiction.Score = 0
Next bcJurisdiction
```

End Sub

Public Function ScoreBest() As cJurisdiction

```
Dim bcJurisdiction As cJurisdiction
Dim lngBestScore As Long
For Each bcJurisdiction In mcolJurisdictions
    If bcJurisdiction.Score > lngBestScore Then
        Set ScoreBest = bcJurisdiction
        lngBestScore = bcJurisdiction.Score
    End If
Next bcJurisdiction
```

End Function

Public Function ScoresBest(Optional MinimumScore As Long, Optional Deviance As Long) As Collection

```
Dim bcJurisdiction As cJurisdiction
Dim lngBestScore As Long
Dim lngMinimum As Long
For Each bcJurisdiction In mcolJurisdictions
    If bcJurisdiction.Score > lngBestScore Then lngBestScore = bcJurisdiction.Score
Next bcJurisdiction

lngMinimum = lngBestScore - Deviance
If MinimumScore > 0 And lngMinimum < MinimumScore Then lngMinimum = MinimumScore

Set ScoresBest = New Collection
If lngBestScore >= lngMinimum Then
    For Each bcJurisdiction In mcolJurisdictions
        If bcJurisdiction.Score >= lngMinimum Then ScoresBest.Add bcJurisdiction
    Next bcJurisdiction
End If
```

End Function

Public Property Get AllCourtGroups() As Collection

```
Set AllCourtGroups = mcolAllCourtGroups
```

End Property

Public Property Get AllCourts() As cCourts

```
Set AllCourts = mbcAllCourts
```

End Property

Public Property Get AllReporters() As cReporters

```
Set AllReporters = mbcAllReporters
```

End Property

Friend Sub Add(Jurisdiction As cJurisdiction, Aliases As Variant)

```
Dim varAlias As Variant
Dim strAliasTrim As String
```

```
With Jurisdiction
```

```
    mcolJurisdictions.Add Jurisdiction, .FullNameTrim
    mbcFuzzyCollection.Add Jurisdiction, .AbrvNameTrim
    If .AbrvNameTrim <> .FullNameTrim Then mbcFuzzyCollection.Add Jurisdiction, .FullNameTrim
End With
```

```

For Each varAlias In Aliases
    strAliasTrim = CStr(varAlias)
    If Not (varAlias = SuperTrim(strAliasTrim)) Then Stop
    mbcFuzzyCollection.Add Jurisdiction, strAliasTrim
Next varAlias

```

```

'-- Aliases should be supplied as trimmed.
'-- For debugging only. Delete later.

```

```
End Sub
```

```
Friend Sub AddAlias(Jurisdiction As cJurisdiction, Alias As String)
```

```
    mbcFuzzyCollection.Add Jurisdiction, Alias
```

```
End Sub
```

```
Public Function ExactItem(FullName As String) As cJurisdiction
```

```
    Set ExactItem = Item(FullName, True, False)
```

```
End Function
```

```
Public Function Item(Key As String, MatchExactAlias As Boolean, MatchSimilarFuzzyCollection As Boolean, Optional KeyIsTrimmed As Boolean = False) As cJurisdiction
```

```

    Dim strKeyTrim As String
    Dim colItems As Collection
    Dim bcReporter As cReporter

```

```

    If KeyIsTrimmed Then
        strKeyTrim = Key
    Else
        strKeyTrim = SuperTrim(Key)
    End If

```

```

    If MatchExactAlias Then Set Item = mbcFuzzyCollection.ExactItem(strKeyTrim)
    If Item Is Nothing And MatchSimilarFuzzyCollection = True Then
        Set colItems = mbcFuzzyCollection.SimilarItems(strKeyTrim)
        If colItems.Count > 0 Then Set Item = colItems(1)
    End If

```

```
End Function
```

```
Public Function Items(Key As String, MatchExactAlias As Boolean, MatchSimilarFuzzyCollection As Boolean, Optional KeyIsTrimmed As Boolean = False) As Collection
```

```

    Dim strKeyTrim As String
    Dim bcReporter As cReporter

```

```

    If KeyIsTrimmed Then
        strKeyTrim = Key
    Else
        strKeyTrim = SuperTrim(Key)
    End If

```

```

    If MatchExactAlias Then Set Items = mbcFuzzyCollection.ExactItems(strKeyTrim)
    If MatchSimilarFuzzyCollection = True Then
        If Items.Count = 0 Then
            Set Items = mbcFuzzyCollection.SimilarItems(strKeyTrim)
        End If
    End If

```

```
End Function
```

```
Public Function Index(Number As Integer) As cJurisdiction
```

```
    Set Index = mcolJurisdictions(Number)
```

```
End Function
```

```
Public Function Count() As Long
```

```
    Count = mcolJurisdictions.Count
```

```
End Function
```

```
Public Function NewEnum() As IUnknown
```

```
    Set NewEnum = mcolJurisdictions.[_NewEnum]
```

```
End Function
```

```
'Public Sub Initialize()
```

```

    Dim bcJurisdiction As cJurisdiction
    Dim bcCourtGroup As cCourtGroup

```

```

    For Each bcJurisdiction In mcolJurisdictions
        For Each bcCourtGroup In bcJurisdiction.CourtGroups
            Call bcCourtGroup.Courts.FuzzyCollection.Initialize
            Call bcCourtGroup.Reporters.FuzzyCollection.Initialize
        Next bcCourtGroup
    Next bcJurisdiction

```

```

    Call mbcFuzzyCollection.Initialize
    Call mbcAllCourts.FuzzyCollection.Initialize
    Call mbcAllReporters.FuzzyCollection.Initialize

```

```
'End Sub
```

```
Dim bcJurisdiction As cJurisdiction
Dim bcCourtGroup As cCourtGroup
Dim bcCourt As cCourt
Dim bcReporter As cReporter
```

```

For Each bcJurisdiction In mcolJurisdictions

```

```
For Each bcCourt In bcJurisdiction.Courts
    Set bcCourt.Parent = Nothing
Next bcCourt
```

```
Set mcolJurisdictions = Nothing
Set mcolAllCourtGroups = Nothing
Set mbcAllCourts = Nothing
Set mbcAllReporters = Nothing
```

1900	1901	1902	1903	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303	2304	2305	2306	2307	2308</
------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	--------

cReporter - 1

'-- Copyright 2001 by Robert Jacobson.

Option Explicit

```
Private mcolEditors As Collection
Private mlngKey As Long
Private mlngVolumeStart As Long
Private mlngVolumeEnd As Long
Private mlngYearStart As Long
Private mlngYearEnd As Long
Private mlngGroupCode As Long
Private mbolJurisdictionImplicit As Boolean
Private mstrFullName As String
Private mstrFullNameTrim As String
Private mstrAbrvName As String
Private mstrAbrvNameTrim As String
Private mbcMainReporter As cReporter
Private mlngMainReporterVolumeFirst As Long
Private mbcImplicitCourt As cCourt
Private mbcParent As cCourtGroup
```

Public Score As Long

```
Public Property Get Editors() As Collection
    Set Editors = mcolEditors
End Property
```

```
Friend Property Get Key() As String
    Key = "K" & mlngKey
End Property
```

```
Public Property Get MainReporter() As cReporter
    Set MainReporter = mbcMainReporter
End Property
```

```
Friend Property Set MainReporter(Source As cReporter)
    Set mbcMainReporter = Source
End Property
```

```
Public Property Get MainReporterVolumeFirst() As Long
    MainReporterVolumeFirst = mlngMainReporterVolumeFirst
End Property
```

```
Friend Property Let MainReporterVolumeFirst(Source As Long)
    mlngMainReporterVolumeFirst = Source
End Property
```

```
Public Function MainReporterVolumeLast() As Long
    If Not (mbcMainReporter Is Nothing) Then
        MainReporterVolumeLast = mlngMainReporterVolumeFirst + (mlngVolumeEnd - mlngVolumeStart)
    End If
End Function
```

```
Public Property Get VolumeStart() As Long
    VolumeStart = mlngVolumeStart
End Property
```

```
Friend Property Let VolumeStart(ByVal Source As Long)
    mlngVolumeStart = Source
End Property
```

```
Public Property Get VolumeEnd() As Long
    VolumeEnd = mlngVolumeEnd
End Property
```

```
Friend Property Let VolumeEnd(ByVal Source As Long)
    mlngVolumeEnd = Source
End Property
```

```
Public Property Get YearStart() As Long
    YearStart = mlngYearStart
End Property
```

```
Friend Property Let YearStart(ByVal Source As Long)
    mlngYearStart = Source
End Property
```

```
Public Property Get YearEnd() As Long
    YearEnd = mlngYearEnd
End Property
```

```
Friend Property Let YearEnd(ByVal Source As Long)
    mlngYearEnd = Source
End Property
```

```
Public Property Get GroupCode() As Long
    GroupCode = mlngGroupCode
End Property
```

```
Friend Property Let GroupCode(ByVal Source As Long)
    mlngGroupCode = Source
End Property
```

```
Public Property Get JurisdictionImplicit() As Boolean
    JurisdictionImplicit = mbolJurisdictionImplicit
End Property
```

```
Friend Property Let JurisdictionImplicit(ByVal Source As Boolean)
    mbolJurisdictionImplicit = Source
End Property
```

cReporter - 2

```

Public Property Get FullName() As String
    FullName = mstrFullName
End Property

Friend Property Let FullName(ByVal Source As String)
    mstrFullName = Source
    mstrFullNameTrim = SuperTrim(Source)
End Property

Public Property Get FullNameTrim() As String
    FullNameTrim = mstrFullNameTrim
End Property

Friend Property Let ABrvName(ByVal Source As String)
    mstrABrvName = Source
    mstrABrvNameTrim = SuperTrim(Source)
End Property

Public Property Get ABrvName() As String
    ABrvName = mstrABrvName
End Property

Public Property Get ABrvNameTrim() As String
    ABrvNameTrim = mstrABrvNameTrim
End Property

Public Function ImplicitJurisdiction() As cJurisdiction
    If mbcJurisdictionImplicit Then
        Set ImplicitJurisdiction = Me.Parent.Parent
    End If
End Function

Public Property Get ImplicitCourt() As cCourt
    Set ImplicitCourt = mbcImplicitCourt
End Property

Friend Property Set ImplicitCourt(Source As cCourt)
    Set mbcImplicitCourt = Source
End Property

Public Property Get Parent() As cCourtGroup
    Set Parent = mbcParent
End Property

Friend Property Set Parent(ByVal Source As cCourtGroup)
    Set mbcParent = Source
End Property

Public Function HasEditors() As Boolean
    If mbcEditors.Count > 0 Then HasEditors = True
End Function

Public Function IsEditorReporter() As Boolean
    If Not (mbcMainReporter Is Nothing) Then IsEditorReporter = True
End Function

Private Sub Class_Initialize()
    mKey = UniqueKey()
    Set mcolEditors = New Collection
    ObjectsCount = ObjectsCount + 1
    Debug.Print "Reporter created. Total: " & ObjectsCount
End Sub

Private Sub Class_Terminate()
    ObjectsCount = ObjectsCount - 1
    Debug.Print "Reporter destroyed. Total: " & ObjectsCount
End Sub

```

cReporters - 1

'-- Copyright 2001 by Robert L. Jacob

Option Explicit

Private mcolReporters As Collection
Private mbcFuzzyCollection As cFuzzyCollection

Friend Property Get FuzzyCollection() As cFuzzyCollection
Set FuzzyCollection = mbcFuzzyCollection
End Property

Private Sub Class_Initialize()
Set mcolReporters = New Collection
Set mbcFuzzyCollection = New cFuzzyCollection

End Sub

Public Sub ScoreReset()

Dim bcReporter As cReporter
For Each bcReporter In mcolReporters
bcReporter.Score = 0
Next bcReporter

End Sub

Public Function ScoreBest() As cReporter

Dim bcReporter As cReporter
Dim lngBestScore As Long
For Each bcReporter In mcolReporters
If bcReporter.Score > lngBestScore Then
Set ScoreBest = bcReporter
lngBestScore = bcReporter.Score
End If
Next bcReporter

End Function

Friend Sub Add(Reporter As cReporter, Optional After As cReporter)

With Reporter
If After Is Nothing Then
mcolReporters.Add Reporter, CStr(.Key)
Else
mcolReporters.Add Reporter, CStr(.Key), After:=CStr(After.Key)
End If
mbcFuzzyCollection.Add Reporter, .AbrvNameTrim
End With

End Sub

Friend Sub AddAlias(Reporter As cReporter, AliasTrim As String)

mbcFuzzyCollection.Add Reporter, AliasTrim

End Sub

Public Function Item(Key As String, MatchExactAlias As Boolean, MatchSimilarAlias As Boolean, Optional ByRef KeyIsTrimmed As Boolean = False) As cReporter

Dim strKeyTrim As String
Dim colItems As Collection
Dim bcReporter As cReporter

If KeyIsTrimmed Then
strKeyTrim = Key
Else
strKeyTrim = SuperTrim(Key)
End If

If MatchExactAlias = False And MatchSimilarAlias = False Then

For Each bcReporter In mcolReporters
If strKeyTrim = bcReporter.AbrvNameTrim Then
Set Item = bcReporter
Exit Function
End If
Next bcReporter

Else
If MatchExactAlias Then Set Item = mbcFuzzyCollection.ExactItem(strKeyTrim)
If Item Is Nothing And MatchSimilarAlias Then
Set colItems = mbcFuzzyCollection.SimilarItems(strKeyTrim)
If colItems.Count > 0 Then Set Item = colItems(1)
End If
End If

End Function

Public Function Items(Key As String, MatchExactAlias As Boolean, MatchSimilarFuzzyCollection As Boolean, Optional ByRef KeyIsTrimmed As Boolean = False) As Collection

Dim strKeyTrim As String
Dim bcReporter As cReporter

If KeyIsTrimmed Then
strKeyTrim = Key

1900	1901	1902	1903	1904	1905	1906	1907	1908	1909	1910	1911	1912	1913	1914	1915	1916	1917	1918	1919	1920	1921	1922	1923	1924	1925	1926	1927	1928	1929	1930	1931	1932	1933	1934	1935	1936	1937	1938	1939	1940	1941	1942	1943	1944	1945	1946	1947	1948	1949	1950	1951	1952	1953	1954	1955	1956	1957	1958	1959	1960	1961	1962	1963	1964	1965	1966	1967	1968	1969	1970	1971	1972	1973	1974	1975	1976	1977	1978	1979	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100	2101	2102	2103	2104	2105	2106	2107	2108	2109	2110	2111	2112	2113	2114	2115	2116	2117	2118	2119	2120	2121	2122	2123	2124	2125	2126	2127	2128	2129	2130	2131	2132	2133	2134	2135	2136	2137	2138	2139	2140	2141	2142	2143	2144	2145	2146	2147	2148	2149	2150	2151	2152	2153	2154	2155	2156	2157	2158	2159	2160	2161	2162	2163	2164	2165	2166	2167	2168	2169	2170	2171	2172	2173	2174	2175	2176	2177	2178	2179	2180	2181	2182	2183	2184	2185	2186	2187	2188	2189	2190	2191	2192	2193	2194	2195	2196	2197	2198	2199	2200	2201	2202	2203	2204	2205	2206	2207	2208	2209	2210	2211	2212	2213	2214	2215	2216	2217	2218	2219	2220	2221	2222	2223	2224	2225	2226	2227	2228	2229	2230	2231	2232	2233	2234	2235	2236	2237	2238	2239	2240	2241	2242	2243	2244	2245	2246	2247	2248	2249	2250	2251	2252	2253	2254	2255	2256	2257	2258	2259	2260	2261	2262	2263	2264	2265	2266	2267	2268	2269	2270	2271	2272	2273	2274	2275	2276	2277	2278	2279	2280	2281	2282	2283	2284	2285	2286	2287	2288	2289	2290	2291	2292	2293	2294	2295	2296	2297	2298	2299	2300	2301	2302	2303	2304	2305	2306	2307	23
------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	----

mGlobals - 1

'-- Copyright 2001 by Robert L. Jackson.

135

Option Explicit

Public gbcJurisdictions As New cJurisdictions
Private mlngUniqueKey As Long

Public Function UniqueKey() As Long

 mlngUniqueKey = mlngUniqueKey + 1
 UniqueKey = mlngUniqueKey

End Function

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
219

'-- Copyright 2001 by Robert L. Jacob

Option Explicit

```
Public ObjectsCount As Long
Public Const bcMINIMUMYEAR = 1600
Public Const bcMAXIMUMYEAR = 2025
```

Public Sub LoadReporters(Jurisdictions As cJurisdictions)

```
    Call LoadIndividualReporters(Jurisdictions)
    Call LoadLexisAndWestlawReporters(Jurisdictions)
```

End Sub

Private Sub LoadIndividualReporters(Jurisdictions As cJurisdictions)

```
Dim lngFileNumber As Long
Dim bcReporter As cReporter
Dim strAbvrName As String
Dim strFullName As String
Dim strMainReporter As String
Dim strJurisdiction As String
Dim strCourtGroup As String
Dim strVolumeStart As String
Dim strVolumeEnd As String
Dim strYearStart As String
Dim strYearEnd As String
Dim strGroupCode As String
Dim strImplicitCourt As String
Dim strMainReporterVolumeFirst As String
Dim bcMainReporter As cReporter
```

```
lngFileNumber = FreeFile
Open "C:\BlueCheck\Data\Reporters.txt" For Input As lngFileNumber
```

```
Do
    Line Input #lngFileNumber, strJurisdiction
    Line Input #lngFileNumber, strCourtGroup
    Line Input #lngFileNumber, strAbvrName
    Line Input #lngFileNumber, strFullName
    Line Input #lngFileNumber, strMainReporter
    Line Input #lngFileNumber, strVolumeStart
    Line Input #lngFileNumber, strVolumeEnd
    Line Input #lngFileNumber, strYearStart
    Line Input #lngFileNumber, strYearEnd
    Line Input #lngFileNumber, strGroupCode
    Line Input #lngFileNumber, strImplicitCourt
    Line Input #lngFileNumber, strMainReporter
    Line Input #lngFileNumber, strMainReporterVolumeFirst

    Set bcReporter = New cReporter
    With bcReporter
        .AbvrName = strAbvrName
        .FullName = strFullName
        .VolumeStart = Val(strVolumeStart)
        .VolumeEnd = Val(strVolumeEnd)
        .YearStart = Val(strYearStart)
        .YearEnd = Val(strYearEnd)
        .GroupCode = strGroupCode
        Set .Parent = Jurisdictions(strJurisdiction).CourtGroups(strCourtGroup)

        If strImplicitCourt <> "" Then
            Set .ImplicitCourt = .Parent.Parent.Courts.Item(strImplicitCourt, False, False)
            .JurisdictionImplicit = True
        ElseIf strJurisdiction = "Federal" Then
            .JurisdictionImplicit = True
        ElseIf InStr(bcReporter.AbrvNameTrim, .Parent.Parent.AbrvNameTrim) Then
            .JurisdictionImplicit = True
        End If

        If strMainReporter <> "" Then
            Set bcMainReporter = .Parent.Reporters(strMainReporter, False, False)
            Set .MainReporter = bcMainReporter
            bcMainReporter.Editors.Add bcReporter
        End If
        .MainReporterVolumeFirst = Val(strMainReporterVolumeFirst)
    End With

    bcReporter.Parent.Reporters.Add bcReporter
    bcReporter.Parent.Reporters.AddAlias bcReporter, bcReporter.FullNameTrim
    bcReporter.Parent.Parent.AllReporters.Add bcReporter
    bcReporter.Parent.Parent.AllReporters.AddAlias bcReporter, bcReporter.FullNameTrim
    Jurisdictions.AllReporters.Add bcReporter
    Jurisdictions.AllReporters.AddAlias bcReporter, bcReporter.FullNameTrim

    DoEvents
Loop Until EOF(lngFileNumber)
Close lngFileNumber
```

End Sub

Private Sub LoadLexisAndWestlawReporters(Jurisdictions As cJurisdictions)

```
Dim bcJurisdiction As cJurisdiction
Dim bcCourtGroup As cCourtGroup
Dim bcReporter As cReporter
```

End Sub

mPopulateJurisdictions - 1

'-- Copyright 2001 by Robert L. Jackson.

138

Option Explicit

Private mbcJurisdictions As cJurisdictions

Public Sub PopulateJurisdictions(Jurisdictions As cJurisdictions)

Set mbcJurisdictions = Jurisdictions

Call AddFederal
Call AddFederalOldCircuitCourts
Call AddAla
Call AddAlaska
Call AddAriz
Call AddArk
Call AddCal
Call AddColo
Call AddConn
Call AddDel
Call AddDC
Call AddFla
Call AddGa
Call AddHaw
Call AddIdaho
Call AddIll
Call AddInd
Call AddIowa
Call AddKan
Call AddKy
Call AddLa
Call AddMe
Call AddMd
Call AddMass
Call AddMich
Call AddMinn
Call AddMiss
Call AddMo
Call AddMont
Call AddNeb
Call AddNev
Call AddNH
Call AddNJ
Call AddNM
Call AddNY
Call AddNC
Call AddND
Call AddOhio
Call AddOkla
Call AddOr
Call AddPa
Call AddRI
Call AddSC
Call AddSD
Call AddTenn
Call AddTex
Call AddUtah
Call AddVt
Call AddVa
Call AddWash
Call AddWVa
Call AddWis
Call AddWyo
Call AddAmSamoa
Call AddCZ
Call AddGuam
Call AddNavajo
Call AddNMariI
Call AddPR
Call AddVI

Set mbcJurisdictions = Nothing

End Sub

Private Sub AddFederal()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
Dim objJurisdiction As cJurisdiction
Dim strJurName As String

Set NewJur = AddJurisdiction("Federal", "Fed.", "us")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Supreme Court", "", 1790, 0, True, "suprct", "supct", "sct")
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 3
NewCourtGroup.NonParallelReporters.Add 4

Set NewCourtGroup = AddCourtGroup(NewJur, "Circuit Justices")
Set NewCourt = AddCourt(NewCourtGroup, "Circuit Justice", "Circuit Justice", 1790, 0, True)
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 3
NewCourtGroup.NonParallelReporters.Add 4

Set NewCourtGroup = AddCourtGroup(NewJur, "Courts of Appeals")

```
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the First Circuit", "1st Cir.", 1891, 0, True, "1cir", "cca1st", "cca1", "ca1", "calst", "cir1", "firstcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Second Circuit", "2d Cir.", 1891, 0, True, "2cir", "cca2d", "cca2", "ca2", "ca2d", "cir2", "secondcir", "2ndcir", "cca2nd", "ca2nd")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Third Circuit", "3d Cir.", 1891, 0, True, "3cir", "cca3d", "cca3", "ca3", "ca3d", "cir3", "thirdcir", "3rdcir", "cca3rd", "ca3rd")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Fourth Circuit", "4th Cir.", 1891, 0, True, "4cir", "cca4th", "cca4", "ca4", "ca4th", "cir4", "fourthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Fifth Circuit", "5th Cir.", 1891, 0, True, "5cir", "cca5th", "cca5", "ca5", "ca5th", "cir5", "fifthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Sixth Circuit", "6th Cir.", 1891, 0, True, "6cir", "cca6th", "cca6", "ca6", "ca6th", "cir6", "sixthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Seventh Circuit", "7th Cir.", 1891, 0, True, "7cir", "cca7th", "cca7", "ca7", "ca7th", "cir7", "seventhcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Eighth Circuit", "8th Cir.", 1891, 0, True, "8cir", "cca8th", "cca8", "ca8", "ca8th", "cir8", "eighthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Ninth Circuit", "9th Cir.", 1891, 0, True, "9cir", "cca9th", "cca9", "ca9", "ca9th", "cir9", "ninthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Tenth Circuit", "10th Cir.", 1891, 0, True, "10cir", "cca10th", "cca10", "ca10", "ca10th", "cir10", "tenthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the Eleventh Circuit", "11th Cir.", 1891, 0, True, "11cir", "cca11th", "cca11", "ca11", "ca11th", "cir11", "eleventhcir")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals for the District of Columbia Circuit", "D.C. Cir.", 1919, 0, True, "ccadc", "cairdc", "cadc", "districtofcolumbiacir")

NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Temporary Emergency Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Temporary Emergency Court of Appeals", "Temp. Emer. Ct. App.", 1971, 1993, True, "tempemerappct")

Set NewCourt = AddCourt(NewCourtGroup, "Emergency Court of Appeals", "Emer. Ct. App.", 1942, 1961, True, "emerappct")
Set NewCourt = AddCourt(NewCourtGroup, "Commerce Court", "Comm. Ct.", 1910, 1913, True)
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "United States Court of Appeals for the Federal Circuit")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Court of Appeals for the Federal Circuit", "Fed. Cir.", 1982, 0, True)
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Court of Customs and Patent Appeals", "C.C.P.A.", 1929, 1982, True)
Set NewCourt = AddCourt(NewCourtGroup, "Court of Customs Appeals", "Ct. Cust. App.", 1910, 1929, True)
Set NewCourt = AddCourt(NewCourtGroup, "Court of Claims", "Ct. Cl.", 1956, 1982, False)
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2

Set NewCourtGroup = AddCourtGroup(NewJur, "United States Court of Federal Claims")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Court of Federal Claims", "Fed. Cl.", 1992, 0, True)
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Claims Court", "Cl. Ct.", 1982, 1992, True)
Set NewCourt = AddCourt(NewCourtGroup, "Court of Claims", "Ct. Cl.", 1863, 1982, True)
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2

Set NewCourtGroup = AddCourtGroup(NewJur, "United States Court of International Trade")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Court of International Trade", "Ct. Int'l Trade", 1980, 0, True)
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Customs Court", "Cust. Ct.", 1926, 1980, True)
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 3
NewCourtGroup.NonParallelReporters.Add 4

Set NewCourtGroup = AddCourtGroup(NewJur, "District Courts")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Alabama", "N.D. Ala.", 1880, 0, True, "ndal", "ndalab")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Middle District of Alabama", "M.D. Ala.", 1880, 0, True, "mdal", "mdalab")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Alabama", "S.D. Ala.", 1880, 0, True, "sdal", "sdalab")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Alaska", "D. Alaska", 1880, 0, True, "dal")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Arizona", "D. Ariz.", 1880, 0, True, "daz", "darizona")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Arkansas", "E.D. Ark.", 1880, 0, True, "edak", "edarka")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Arkansas", "W.D. Ark.", 1880, 0, True, "wdak", "wdarka")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of California", "N.D. Cal.", 1880, 0, True, "ndca", "ndcalifornia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of California", "E.D. Cal.", 1880, 0, True, "edca", "edcalifornia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Central District of California", "C.D. Cal.", 1880, 0, True, "cdca", "cdcalifornia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of California", "S.D. Cal.", 1880, 0, True, "sdca", "sdcalifornia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Colorado", "D. Colo.", 1880, 0, True, "dco", "dcolorado")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Connecticut", "D. Conn.", 1880, 0, True, "dct", "dconnecticut")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Delaware", "D. Del.", 1880, 0, True, "dde", "ddelaware")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of District of Columbia", "D.D.C.", 1880, 0, True, "ddistrictofcolumbia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Florida", "N.D. Fla.", 1880, 0, True, "ndfl", "ndflorida")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Middle District of Florida", "M.D. Fla.", 1880, 0, True, "mdfl", "mdflorida")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Florida", "S.D. Fla.", 1880, 0, True, "sdfl", "sdflorida")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Georgia", "N.D. Ga.", 1880, 0, True, "ndgeorgia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Middle District of Georgia", "M.D. Ga.", 1880, 0, True, "mdgeorgia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Georgia", "S.D. Ga.", 1880, 0, True, "sdgeorgia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Hawaii", "D. Haw.", 1880, 0, True, "dhi", "dhawaii")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Idaho", "D. Idaho", 1880, 0, True, "did", "didaho")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Illinois", "N.D. Ill.", 1880, 0, True, "ndil", "ndillinois")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Central District of Illinois", "C.D. Ill.", 1880, 0, True, "cdil", "cdillinois")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Illinois", "S.D. Ill.", 1880, 0, True, "sdil", "sdillinois")
```

```

Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Indiana", "N.D. Ind.", 1880, 0, True, "ndin", "ndindi
ana")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Indiana", "S.D. Ind.", 1880, 0, True, "sdin", "sdindi
ana")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Iowa", "N.D. Iowa", 1880, 0, True, "ndia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Iowa", "S.D. Iowa", 1880, 0, True, "sdia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Kansas", "D. Kan.", 1880, 0, True, "dks", "dkansas")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Kentucky", "E.D. Ky.", 1880, 0, True, "edkentucky")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Kentucky", "W.D. Ky.", 1880, 0, True, "wdkentucky")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Louisiana", "E.D. La.", 1880, 0, True, "edlouisiana")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Middle District of Louisiana", "M.D. La.", 1880, 0, True, "mdlouisiana")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Louisiana", "W.D. La.", 1880, 0, True, "wdlouisiana")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Maine", "D. Me.", 1880, 0, True, "dmaine")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Maryland", "D. Md.", 1880, 0, True, "dmaryland")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Massachusetts", "D. Mass.", 1880, 0, True, "dma", "dmassachuse
ts")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Michigan", "E.D. Mich.", 1880, 0, True, "edmi", "edmic
higan")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Michigan", "W.D. Mich.", 1880, 0, True, "wdmi", "wdmic
higan")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Minnesota", "D. Minn.", 1880, 0, True, "dmn", "dminnesota")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Mississippi", "N.D. Miss.", 1880, 0, True, "ndmi", "n
dmississippi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Mississippi", "S.D. Miss.", 1880, 0, True, "sdmi", "s
dmississippi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Missouri", "E.D. Mo.", 1880, 0, True, "edmissouri")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Missouri", "W.D. Mo.", 1880, 0, True, "wdmissouri")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Montana", "D. Mont.", 1880, 0, True, "dmt", "dmontana")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Nebraska", "D. Neb.", 1880, 0, True, "dne", "dnebraska")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Nevada", "D. Nev.", 1880, 0, True, "dnv", "dnevada")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of New Hampshire", "D.N.H.", 1880, 0, True, "dnewhampshire")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of New Jersey", "D.N.J.", 1880, 0, True, "dnewjersey")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of New Mexico", "D.N.M.", 1880, 0, True, "dnewmexico")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of New York", "N.D.N.Y.", 1880, 0, True, "ndnewyork")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of New York", "S.D.N.Y.", 1880, 0, True, "sdnewyork")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of New York", "E.D.N.Y.", 1880, 0, True, "ednewyork")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of New York", "W.D.N.Y.", 1880, 0, True, "wdnewyork")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of North Carolina", "E.D.N.C.", 1880, 0, True, "ednorthca
rolina")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of North Carolina", "W.D.N.C.", 1880, 0, True, "wdnorthca
rolina")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Middle District of North Carolina", "M.D.N.C.", 1880, 0, True, "mdnorthcar
olina")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of North Dakota", "D.N.D.", 1880, 0, True, "dnorthdakota")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Ohio", "N.D. Ohio", 1880, 0, True, "ndoh")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Ohio", "S.D. Ohio", 1880, 0, True, "sdoh")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Oklahoma", "N.D. Okla.", 1880, 0, True, "ndok", "ndok
lahoma")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Oklahoma", "E.D. Okla.", 1880, 0, True, "edok", "edokl
ahoma")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Oklahoma", "W.D. Okla.", 1880, 0, True, "wdok", "wdokl
ahoma")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Oregon", "D. Or.", 1880, 0, True, "doregon")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Pennsylvania", "E.D. Pa.", 1880, 0, True, "edpennsylv
nia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Middle District of Pennsylvania", "M.D. Pa.", 1880, 0, True, "mdpennsylvan
ia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Pennsylvania", "W.D. Pa.", 1880, 0, True, "wdpennsylvan
ia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Rhode Island", "D.R.I.", 1880, 0, True, "drhodeisland")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of South Carolina", "D.S.C.", 1880, 0, True, "dsouthcarolina")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of South Dakota", "D.S.D.", 1880, 0, True, "dsouthdakota")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Tennessee", "E.D. Tenn.", 1880, 0, True, "edtn", "edte
nnessee")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Middle District of Tennessee", "M.D. Tenn.", 1880, 0, True, "mdtn", "mdten
nessee")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Tennessee", "W.D. Tenn.", 1880, 0, True, "wdtn", "wdte
nnessee")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of Texas", "N.D. Tex.", 1880, 0, True, "ndtx", "ndtexas")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Texas", "E.D. Tex.", 1880, 0, True, "edtx", "edtexas")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of Texas", "S.D. Tex.", 1880, 0, True, "sdtx", "sdtexas")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Texas", "W.D. Tex.", 1880, 0, True, "wdtx", "wdtexas")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Utah", "D. Utah", 1880, 0, True, "dut", "dutah")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Vermont", "D. Vt.", 1880, 0, True, "dvermont")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Virginia", "E.D. Va.", 1880, 0, True, "edvirginia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Virginia", "W.D. Va.", 1880, 0, True, "wdvirginia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Washington", "E.D. Wash.", 1880, 0, True, "edwa", "edw
n", "edwashington")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Washington", "W.D. Wash.", 1880, 0, True, "wdwa", "wdw
n", "wdwashington")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Northern District of West Virginia", "N.D.W. Va.", 1880, 0, True, "ndwv",
"ndwestvirginia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Southern District of West Virginia", "S.D.W. Va.", 1880, 0, True, "sdwv",
"sdwestvirginia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Eastern District of Wisconsin", "E.D. Wis.", 1880, 0, True, "edwi", "edwis
consin")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, Western District of Wisconsin", "W.D. Wis.", 1880, 0, True, "wdwi", "wdwis
consin")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court, District of Wyoming", "D. Wyo.", 1880, 0, True, "dwy", "dwyoming")
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2

Set NewCourtGroup = AddCourtGroup(mbcJurisdictions("Federal"), "Bankruptcy Courts")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Alabama", "Bankr. N.D. Ala.", 1979, 0, True, "bankr
ndalabama", "bnkrndala", "bnkrndal", "bnkrndal")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Middle District of Alabama", "Bankr. M.D. Ala.", 1979, 0, True, "bankrmd
alabama", "bnkrmdala", "bnkrmdal", "bnkrmdal")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Alabama", "Bankr. S.D. Ala.", 1979, 0, True, "bankr
sdalabama", "bnkrsdala", "bnkrsdal", "bnkrsdal")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Alaska", "Bankr. D. Alaska", 1979, 0, True, "bnkrdalaska", "
bankrdal", "bnkrdal")

```

```
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Arizona", "Bankr. D. Ariz.", 1979, 0, True, "bnkrdariz", "bnkrdariz", "bnkrdariz")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Arkansas", "Bankr. E.D. Ark.", 1979, 0, True, "bnkrredarkansas", "bnkrredark", "bnkrredak", "bnkrredak")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Arkansas", "Bankr. W.D. Ark.", 1979, 0, True, "bnkrwdarkansas", "bnkrwdark", "bnkrwdak", "bnkrwdak")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of California", "Bankr. N.D. Cal.", 1979, 0, True, "bnkrndcalifornia", "bnkrndcal", "bnkrndca", "bnkrndca")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of California", "Bankr. E.D. Cal.", 1979, 0, True, "bnkrndcalifornia", "bnkrndcal", "bnkrndca", "bnkrndca")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Central District of California", "Bankr. C.D. Cal.", 1979, 0, True, "bnkrndcalifornia", "bnkrndcal", "bnkrndca", "bnkrndca")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of California", "Bankr. S.D. Cal.", 1979, 0, True, "bnkrndcalifornia", "bnkrndcal", "bnkrndca", "bnkrndca")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Colorado", "Bankr. D. Colo.", 1979, 0, True, "bnkrndcolorado", "bnkrndcolo", "bnkrndco", "bnkrndco")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Connecticut", "Bankr. D. Conn.", 1979, 0, True, "bnkrndconnecticut", "bnkrndconn", "bnkrndct", "bnkrndct")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Delaware", "Bankr. D. Del.", 1979, 0, True, "bnkrnddelaware", "bnkrnddel", "bnkrndde", "bnkrndde")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of District of Columbia", "Bankr. D.D.C.", 1979, 0, True, "bnkrnddistrictofcolumbia", "bnkrnddc", "bnkrnddc", "bnkrnddc")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Florida", "Bankr. N.D. Fla.", 1979, 0, True, "bnkrndflorida", "bnkrndfla", "bnkrndfl", "bnkrndfl")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Middle District of Florida", "Bankr. M.D. Fla.", 1979, 0, True, "bnkrndflorida", "bnkrndfla", "bnkrndfl", "bnkrndfl")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Florida", "Bankr. S.D. Fla.", 1979, 0, True, "bnkrndflorida", "bnkrndfla", "bnkrndfl", "bnkrndfl")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Georgia", "Bankr. N.D. Ga.", 1979, 0, True, "bnkrndgeorgia", "bnkrndga", "bnkrndga", "bnkrndga")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Middle District of Georgia", "Bankr. M.D. Ga.", 1979, 0, True, "bnkrndgeorgia", "bnkrndga", "bnkrndga", "bnkrndga")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Georgia", "Bankr. S.D. Ga.", 1979, 0, True, "bnkrndgeorgia", "bnkrndga", "bnkrndga", "bnkrndga")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Hawaii", "Bankr. D. Haw.", 1979, 0, True, "bnkrndhawaii", "bnkrndhaw", "bnkrndhi", "bnkrndhi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Idaho", "Bankr. D. Idaho", 1979, 0, True, "bnkrndidaho", "bnkrndid", "bnkrndid", "bnkrndid")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Illinois", "Bankr. N.D. Ill.", 1979, 0, True, "bnkrndillinois", "bnkrndill", "bnkrndil", "bnkrndil")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Central District of Illinois", "Bankr. C.D. Ill.", 1979, 0, True, "bnkrndillinois", "bnkrndill", "bnkrndil", "bnkrndil")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Illinois", "Bankr. S.D. Ill.", 1979, 0, True, "bnkrndillinois", "bnkrndill", "bnkrndil", "bnkrndil")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Indiana", "Bankr. N.D. Ind.", 1979, 0, True, "bnkrndindiana", "bnkrndind", "bnkrndin", "bnkrndin")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Indiana", "Bankr. S.D. Ind.", 1979, 0, True, "bnkrndindiana", "bnkrndind", "bnkrndin", "bnkrndin")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Iowa", "Bankr. N.D. Iowa", 1979, 0, True, "bnkrndiowa", "bnkrndia", "bnkrndia", "bnkrndia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Iowa", "Bankr. S.D. Iowa", 1979, 0, True, "bnkrndiowa", "bnkrndia", "bnkrndia", "bnkrndia")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Kansas", "Bankr. D. Kan.", 1979, 0, True, "bnkrndkansas", "bnkrndkan", "bnkrndks", "bnkrndks")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Kentucky", "Bankr. E.D. Ky.", 1979, 0, True, "bnkrndkentucky", "bnkrndken", "bnkrndky", "bnkrndky")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Kentucky", "Bankr. W.D. Ky.", 1979, 0, True, "bnkrndkentucky", "bnkrndken", "bnkrndky", "bnkrndky")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Louisiana", "Bankr. E.D. La.", 1979, 0, True, "bnkrndlouisiana", "bnkrndlea", "bnkrndla", "bnkrndla")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Middle District of Louisiana", "Bankr. M.D. La.", 1979, 0, True, "bnkrndlouisiana", "bnkrndlea", "bnkrndla", "bnkrndla")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Louisiana", "Bankr. W.D. La.", 1979, 0, True, "bnkrndlouisiana", "bnkrndlea", "bnkrndla", "bnkrndla")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Maine", "Bankr. D. Me.", 1979, 0, True, "bnkrndmaine", "bnkrndme", "bnkrndme", "bnkrndme")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Maryland", "Bankr. D. Md.", 1979, 0, True, "bnkrndmaryland", "bnkrndmd", "bnkrndmd", "bnkrndmd")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Massachusetts", "Bankr. D. Mass.", 1979, 0, True, "bnkrndmassachusetts", "bnkrndmas", "bnkrndma", "bnkrndma")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Michigan", "Bankr. E.D. Mich.", 1979, 0, True, "bnkrndmichigan", "bnkrndmich", "bnkrndmi", "bnkrndmi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Michigan", "Bankr. W.D. Mich.", 1979, 0, True, "bnkrndmichigan", "bnkrndmich", "bnkrndmi", "bnkrndmi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Minnesota", "Bankr. D. Minn.", 1979, 0, True, "bnkrndminnesota", "bnkrndminn", "bnkrndmn", "bnkrndmn")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Mississippi", "Bankr. N.D. Miss.", 1979, 0, True, "bnkrndmississippi", "bnkrndmiss", "bnkrndmi", "bnkrndmi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Mississippi", "Bankr. S.D. Miss.", 1979, 0, True, "bnkrndmississippi", "bnkrndmiss", "bnkrndmi", "bnkrndmi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Missouri", "Bankr. E.D. Mo.", 1979, 0, True, "bnkrndmissouri", "bnkrndmo", "bnkrndmo", "bnkrndmo")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Missouri", "Bankr. W.D. Mo.", 1979, 0, True, "bnkrndmissouri", "bnkrndmo", "bnkrndmo", "bnkrndmo")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Montana", "Bankr. D. Mont.", 1979, 0, True, "bnkrndmontana", "bnkrndmt", "bnkrndmt", "bnkrndmt")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Nebraska", "Bankr. D. Neb.", 1979, 0, True, "bnkrndnebraska", "bnkrndne", "bnkrndneb", "bnkrndneb")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Nevada", "Bankr. D. Nev.", 1979, 0, True, "bnkrndnevada", "bnkrndnv", "bnkrndnev", "bnkrndnev")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of New Hampshire", "Bankr. D.N.H.", 1979, 0, True, "bnkrndnewhampshire", "bnkrndnh", "bnkrndnh", "bnkrndnh")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of New Jersey", "Bankr. D.N.J.", 1979, 0, True, "bnkrndnewjersey", "bnkrndnj", "bnkrndnj", "bnkrndnj")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of New Mexico", "Bankr. D.N.M.", 1979, 0, True, "bnkrndnewmexico", "bnkrndnm", "bnkrndnm", "bnkrndnm")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of New York", "Bankr. N.D.N.Y.", 1979, 0, True, "bnkrndnewyork", "bnkrndny", "bnkrndny", "bnkrndny")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of New York", "Bankr. S.D.N.Y.", 1979, 0, True, "bnkrndnewyork", "bnkrndny", "bnkrndny", "bnkrndny")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of New York", "Bankr. E.D.N.Y.", 1979, 0, True, "bnkrndnewyork", "bnkrndny", "bnkrndny", "bnkrndny")
```

dnewyork", "bnkredny")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of New York", "Bankr. W.D.N.Y.", 1979, 0, True, "bankrw
dnewyork", "bnkrdwny")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of North Carolina", "Bankr. E.D.N.C.", 1979, 0, True, "b
bankrednorthcarolina", "bnkrednc")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of North Carolina", "Bankr. W.D.N.C.", 1979, 0, True, "b
bankrwdnorthcarolina", "bnkrwdnc")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Middle District of North Carolina", "Bankr. M.D.N.C.", 1979, 0, True, "b
ankrmdnorthcarolina", "bnkrmdnc")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of North Dakota", "Bankr. D.N.D.", 1979, 0, True, "bankrdnorthd
akota", "bnkrdnd")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Ohio", "Bankr. N.D. Ohio", 1979, 0, True, "bnkrndoh
io", "bnkrndoh")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Ohio", "Bankr. S.D. Ohio", 1979, 0, True, "bnkrsdoh
io", "bnkrsdoh")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Oklahoma", "Bankr. N.D. Okla.", 1979, 0, True, "ban
krndoklahoma", "bnkrndokla", "bnkrndok", "bnkrndok")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Oklahoma", "Bankr. E.D. Okla.", 1979, 0, True, "bank
redoklahoma", "bnkredokla", "bnkredok", "bnkredok")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Oklahoma", "Bankr. W.D. Okla.", 1979, 0, True, "bank
rdoklahoma", "bnkrwdokla", "bnkrwdok", "bnkrwdok")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Oregon", "Bankr. D. Or.", 1979, 0, True, "bankrdoregon", "bn
krdor")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Pennsylvania", "Bankr. E.D. Pa.", 1979, 0, True, "ba
nkredpennsylvania", "bnkredpa")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Middle District of Pennsylvania", "Bankr. M.D. Pa.", 1979, 0, True, "ban
krmdpennsylvania", "bnkrmdpa")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Pennsylvania", "Bankr. W.D. Pa.", 1979, 0, True, "ba
nkrwdpennsylvania", "bnkrwdpa")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Rhode Island", "Bankr. D.R.I.", 1979, 0, True, "bankrdrhodei
sland", "bnkrdr")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of South Carolina", "Bankr. D.S.C.", 1979, 0, True, "bankrdsout
hcarolina", "bnkrdsc")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of South Dakota", "Bankr. D.S.D.", 1979, 0, True, "banksouthda
kota", "bnkrdsd")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Tennessee", "Bankr. E.D. Tenn.", 1979, 0, True, "ban
kredtennessee", "bnkredtenn", "bnkredtn", "bnkredtn")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Middle District of Tennessee", "Bankr. M.D. Tenn.", 1979, 0, True, "bank
rwdtennessee", "bnkrmdtenn", "bnkrmdtn", "bnkrmdtn")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Tennessee", "Bankr. W.D. Tenn.", 1979, 0, True, "ban
krwdtennessee", "bnkrwdtenn", "bnkrwdtn", "bnkrwdtn")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of Texas", "Bankr. N.D. Tex.", 1979, 0, True, "bankrndt
exas", "bnkrndtex", "bnkrndtx", "bnkrndtx")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Texas", "Bankr. E.D. Tex.", 1979, 0, True, "bankredt
exas", "bnkredtex", "bnkredtx", "bnkredtx")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of Texas", "Bankr. S.D. Tex.", 1979, 0, True, "banksrdt
exas", "bnkrsdtx", "bnkrsdtx", "bnkrsdtx")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Texas", "Bankr. W.D. Tex.", 1979, 0, True, "bankrwdt
exas", "bnkrwdtex", "bnkrwdtx", "bnkrwdtx")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Utah", "Bankr. D. Utah", 1979, 0, True, "bnkrdutah", "bnkrd
ut", "bnkrdut")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Vermont", "Bankr. D. Vt.", 1979, 0, True, "bankrdvermont", "b
bnkrdvt")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Virginia", "Bankr. E.D. Va.", 1979, 0, True, "bankre
dvirginia", "bnkredva")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Virginia", "Bankr. W.D. Va.", 1979, 0, True, "bankrw
dvirginia", "bnkrwdva")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Washington", "Bankr. E.D. Wash.", 1979, 0, True, "ba
nkredwashington", "bnkredwash", "bnkredwa", "bnkredwn", "bnkredwn")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Washington", "Bankr. W.D. Wash.", 1979, 0, True, "ba
nkrwdwashington", "bnkrwdwash", "bnkrwdwa", "bnkrwdwn", "bnkrwdwn")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Northern District of West Virginia", "Bankr. N.D.W. Va.", 1979, 0, True, "bank
bnkrndwestvirginia", "bnkrndwva", "bnkrndwv", "bnkrndwv")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Southern District of West Virginia", "Bankr. S.D.W. Va.", 1979, 0, True, "bank
bnkrswdwestvirginia", "bnkrswdwa", "bnkrswdw", "bnkrswdw")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Eastern District of Wisconsin", "Bankr. E.D. Wis.", 1979, 0, True, "bank
redwisconsin", "bnkredwis", "bnkredwi", "bnkredwi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, Western District of Wisconsin", "Bankr. W.D. Wis.", 1979, 0, True, "bank
rdwisconsin", "bnkrwdwis", "bnkrwdwi", "bnkrwdwi")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Bankruptcy Court, District of Wyoming", "Bankr. D. Wyo.", 1979, 0, True, "dwyoming", "bnkr
dwyo", "bnkrdwy", "bnkrdwy")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the First Circuit", "B.A.P. 1st Cir.", 1979, 0, True, "bankrapp
1stcir", "bnkrapp1stcir", "bnkr1stcir", "bnkr1stcir", "bap1cir", "bapccalst", "bapccal", "bapcal", "bapcalst", "bapcir1", "bapfirstcir")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Second Circuit", "B.A.P. 2d Cir.", 1979, 0, True, "bankrapp
2dcir", "bnkrapp2dcir", "bnkr2dcir", "bnkr2dcir", "bnkr2dcir", "bnkr2dcir", "bap2cir", "bapcca2d", "bapcca2", "bapca2", "bapca2d", "bapcir2", "ba
psecondcir", "bap2ndcir", "bapcca2nd", "bapcca2nd")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Third Circuit", "B.A.P. 3d Cir.", 1979, 0, True, "bankrapp
3dcir", "bnkrapp3dcir", "bnkr3dcir", "bnkr3dcir", "bnkr3dcir", "bnkr3dcir", "bap3cir", "bapcca3d", "bapcca3", "bapca3", "bapca3d", "bapcir3", "bap
thirdcir", "bap3rdcir", "bapcca3rd", "bapcca3rd")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Fourth Circuit", "B.A.P. 4th Cir.", 1979, 0, True, "bankrapp
p4thcir", "bnkrapp4thcir", "bnkr4thcir", "bnkr4thcir", "bap4cir", "bapcca4th", "bapcca4", "bapca4", "bapca4th", "bapcir4", "bapfourthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Fifth Circuit", "B.A.P. 5th Cir.", 1979, 0, True, "bankrapp
5thcir", "bnkrapp5thcir", "bnkr5thcir", "bnkr5thcir", "bap5cir", "bapcca5th", "bapcca5", "bapca5", "bapca5th", "bapcir5", "bapfifthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Sixth Circuit", "B.A.P. 6th Cir.", 1979, 0, True, "bankrapp
6thcir", "bnkrapp6thcir", "bnkr6thcir", "bnkr6thcir", "bap6cir", "bapcca6th", "bapcca6", "bapca6", "bapca6th", "bapcir6", "bapsixthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Seventh Circuit", "B.A.P. 7th Cir.", 1979, 0, True, "bankrap
pp7thcir", "bnkrapp7thcir", "bnkr7thcir", "bnkr7thcir", "bap7cir", "bapcca7th", "bapcca7", "bapca7", "bapca7th", "bapcir7", "bapseventhci
r")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Eighth Circuit", "B.A.P. 8th Cir.", 1979, 0, True, "bankrapp
p8thcir", "bnkrapp8thcir", "bnkr8thcir", "bnkr8thcir", "bap8cir", "bapcca8th", "bapcca8", "bapca8", "bapca8th", "bapcir8", "bapeighthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Ninth Circuit", "B.A.P. 9th Cir.", 1979, 0, True, "bankrapp
9thcir", "bnkrapp9thcir", "bnkr9thcir", "bnkr9thcir", "bap9cir", "bapcca9th", "bapcca9", "bapca9", "bapca9th", "bapcir9", "bapninthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Tenth Circuit", "B.A.P. 10th Cir.", 1979, 0, True, "bankrap
p10thcir", "bnkrapp10thcir", "bnkr10thcir", "bnkr10thcir", "bap10cir", "bapcca10th", "bapcca10", "bapca10", "bapca10th", "bapcir10", "bap
tenthcir")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the Eleventh Circuit", "B.A.P. 11th Cir.", 1979, 0, True, "bankr
app11thcir", "bnkrapp11thcir", "bnkr11thcir", "bnkr11thcir", "bap11cir", "bapcca11th", "bapcca11", "bapca11", "bapca11th", "bapcir11", "b
bapeleventhcir")
Set NewCourt = AddCourt(NewCourtGroup, "Bankruptcy Appellate Panel for the D.C. Circuit", "B.A.P. D.C. Cir.", 1979, 0, True, "bankrapp

```

dcccir", "bnkrappppdcccir", "bankrdcccir", "bnkr", "bapccadc", "bapcirdc", "bapdist", "biacir")
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Judicial Panel on Multidistrict Litigation")
Set NewCourt = AddCourt(NewCourtGroup, "Judicial Panel on Multidistrict Litigation", "J.P.M.L.", 1968, 0, True)
Set NewCourt = AddCourt(NewCourtGroup, "Special Court, Regional Rail Reorganization Act", "Regional Rail Reorg. Ct.", 1973, 0, True)
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Tax Court")
Set NewCourt = AddCourt(NewCourtGroup, "Tax Court", "T.C.", 1942, 0, True, "taxct")
Set NewCourt = AddCourt(NewCourtGroup, "Board of Tax Appeals", "B.T.A.", 1924, 1942, True, "bdtaxapp", "boardtaxapp")
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2

Set NewCourtGroup = AddCourtGroup(NewJur, "United States Court of Veterans Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Court of Veterans Appeals", "Vet. App.", 1988, 0, True, "vetctapp", "vetappct")
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "United States Court of Appeals for the Armed Forces")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Court of Appeals for the Armed Forces", "C.A.A.F.", 1975, 0, True, "ctapparmedforces", "ctappaf", "ctapparmedf", "appctarmedforces", "appctaf")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. Court of Military Appeals", "C.M.A.", 1951, 1975, True, "ctmilapp", "ctmilitaryapp")
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

Set NewCourtGroup = AddCourtGroup(NewJur, "Courts of Military Review")
Set NewCourt = AddCourt(NewCourtGroup, "Courts of Military Review", "A.C.M.R.", 0, 0, True)
Set NewCourt = AddCourt(NewCourtGroup, "Courts of Military Review", "A.F.C.M.R.", 0, 0, True)
Set NewCourt = AddCourt(NewCourtGroup, "Courts of Military Review", "C.G.C.M.R.", 0, 0, True)
Set NewCourt = AddCourt(NewCourtGroup, "Courts of Military Review", "N-M.C.M.R.", 0, 0, True)
Set NewCourt = AddCourt(NewCourtGroup, "Boards of Review", "A.B.R.", 0, 0, True)
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2

End Sub

Private Sub AddFederalOldCircuitCourts()

Dim NewCourt As cCourt
Dim NewCourtGroup As cCourtGroup
Dim bcCourt As cCourt
Dim strFullName As String

Set NewCourtGroup = AddCourtGroup(mbcJurisdictions("Federal"), "Circuit Courts")

For Each bcCourt In mbcJurisdictions("Federal").Courts
If bcCourt.CourtGroup.FullName = "District Courts" Then
strFullName = "U.S. Circuit Court" & Right$(bcCourt.FullName, Len(bcCourt.FullName) - 19)
Set NewCourt = AddCourt(NewCourtGroup, strFullName, "C.C." & bcCourt.AbrvName, 0, 1912, True)
End If
Next bcCourt
NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddAla()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Alabama", "Ala.", "al")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Alabama Supreme Court", "", 1840, 0, False, "suprct", "supct", "sct")

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Civil Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Alabama Court of Civil Appeals", "Civ. App.", 1969, 0, False, "appciv")
Set NewCourt = AddCourt(NewCourtGroup, "Alabama Court of Criminal Appeals", "Crim. App.", 1969, 0, False, "appccrim")
Set NewCourt = AddCourt(NewCourtGroup, "Alabama Court of Appeals", "Ct. App.", 1910, 1968, False, "appct", "app")

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddAlaska()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Alaska", "Alaska", "ak")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Alaska Supreme Court", "", 1960, 0, False, "suprct", "supct", "sct")
NewCourtGroup.NonParallelReporters.Add 2

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Alaska Court of Appeals", "Ct. App.", 1980, 0, False, "appct", "app")
NewCourtGroup.NonParallelReporters.Add 2

Set NewCourtGroup = AddCourtGroup(NewJur, "District Courts of Alaska")
Set NewCourt = AddCourt(NewCourtGroup, "District Courts of Alaska", "D. Alaska", 1884, 1959, True, "dal")

```



```
NewCourtGroup.ParallelReporters.Add 2
NewCourtGroup.ParallelReporters.Add 2
NewCourtGroup.ParallelReporters.Add 3
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "United States District Courts")
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Courts for California", "D. Cal.", 1867, 1884, True)
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Courts for Oregon", "D. Or.", 1867, 1884, True)
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Courts of Washington", "D. Wash.", 1867, 1884, True)
```

```
NewCourtGroup.ParallelReporters.Add 2
NewCourtGroup.ParallelReporters.Add 3
```

```
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 2
```

End Sub

Private Sub AddAriz()

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Arizona", "Ariz.", "az")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Arizona Supreme Court", "", 1866, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Arizona Court of Appeals", "Ct. App.", 1965, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

End Sub

Private Sub AddArk()

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Arkansas", "Ark.", "ar")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Arkansas Supreme Court", "", 1837, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Arkansas Court of Appeals", "Ct. App.", 1979, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

End Sub

Private Sub AddCal()

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("California", "Cal.", "ca")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "California Supreme Court", "", 1850, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
NewCourtGroup.ParallelReporters.Add 3
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3
NewCourtGroup.NonParallelReporters.Add 4
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeal")
Set NewCourt = AddCourt(NewCourtGroup, "California Court of Appeals", "Ct. App.", 1905, 0, False, "appct", "app")
Set NewCourt = AddCourt(NewCourtGroup, "California District Court of Appeal", "Dist. Ct. App.", 1905, 0, False, "distappct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 3
```

```
NewCourtGroup.NonParallelReporters.Add 3
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Appellate Departments of the Superior Court")
Set NewCourt = AddCourt(NewCourtGroup, "California Appellate Departments of the Superior Court", "App. Dep't Super. Ct.", 1929, 0, False)
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 3
```

```
NewCourtGroup.NonParallelReporters.Add 3
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddColo()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Colorado", "Colo.", "co")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Colorado Supreme Court", "", 1864, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Colorado Court of Appeals", "Ct. App.", 1891, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3
```

```
End Sub
```

```
Private Sub AddConn()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Connecticut", "Conn.", "ct")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Connecticut Supreme Court", "", 1789, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Appellate Court")
Set NewCourt = AddCourt(NewCourtGroup, "Connecticut Appellate Court", "App. Ct.", 1983, 0, False, "ctapp", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Superior Court")
Set NewCourt = AddCourt(NewCourtGroup, "Connecticut Superior Court", "Super. Ct.", 1935, 0, False, "supct")
Set NewCourt = AddCourt(NewCourtGroup, "Connecticut Court of Common Pleas", "C.P.", 1935, 0, False)
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Circuit Court")
Set NewCourt = AddCourt(NewCourtGroup, "Connecticut Circuit Court", "Cir. Ct.", 1961, 0, False)
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddDel()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
```

```

Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Delaware", "Del.", "de")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Delaware Supreme Court", "", 1792, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Chancery")
Set NewCourt = AddCourt(NewCourtGroup, "Delaware Court of Chancery", "Ch.", 1814, 0, False)

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Superior Court")
Set NewCourt = AddCourt(NewCourtGroup, "Delaware Superior Court", "Super. Ct.", 1951, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "Delaware Superior Court and Orphans' Court", "Super. Ct. & Orphans' Ct.", 1951, 0, False)

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Family Court")
Set NewCourt = AddCourt(NewCourtGroup, "Delaware Family Court", "Fam. Ct.", 1977, 0, False)
NewCourtGroup.NonParallelReporters.Add 2

End Sub

Private Sub AddDC()
    Dim NewJur As cJurisdiction
    Dim NewCourtGroup As cCourtGroup
    Dim NewCourt As cCourt

    Set NewJur = AddJurisdiction("District of Columbia", "D.C.")

    Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
    Set NewCourt = AddCourt(NewCourtGroup, "District of Columbia Court of Appeals", "", 1943, 0, False, "ctapp", "app", "appct", "app")
    Set NewCourt = AddCourt(NewCourtGroup, "District of Columbia Municipal Court of Appeals", "", 1943, 0, False, "munctapp", "munappct")
    NewCourtGroup.NonParallelReporters.Add 2

    Set NewCourtGroup = AddCourtGroup(NewJur, "United States Court of Appeals for the District of Columbia")
    Set NewCourt = AddCourt(NewCourtGroup, "U.S. Court of Appeals for the District of Columbia Circuit", "Cir.", 0, 0, False, "cca", "cir", "ca")
    Set NewCourt = AddCourt(NewCourtGroup, "Supreme Court of the District of Columbia", "", 1841, 1942, False)

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 3

End Sub

Private Sub AddFla()
    Dim NewJur As cJurisdiction
    Dim NewCourtGroup As cCourtGroup
    Dim NewCourt As cCourt

    Set NewJur = AddJurisdiction("Florida", "Fla.", "fl")

    Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
    Set NewCourt = AddCourt(NewCourtGroup, "Florida Supreme Court", "", 1846, 0, False, "suprct", "supct", "sct")
    NewCourtGroup.ParallelReporters.Add 1
    NewCourtGroup.ParallelReporters.Add 2

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 1
    NewCourtGroup.NonParallelReporters.Add 3

    Set NewCourtGroup = AddCourtGroup(NewJur, "District Court of Appeal")
    Set NewCourt = AddCourt(NewCourtGroup, "Florida District Court of Appeal", "Dist. Ct. App.", 1957, 0, False, "distappct")

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 3

    Set NewCourtGroup = AddCourtGroup(NewJur, "Circuit Court")
    Set NewCourt = AddCourt(NewCourtGroup, "Florida Circuit Court", "Cir. Ct.", 0, 0, False)
    Set NewCourt = AddCourt(NewCourtGroup, "Florida County Court", "County Ct.", 0, 0, False)
    Set NewCourt = AddCourt(NewCourtGroup, "Florida Public Service Commission", "P.S.C.", 0, 0, False)

    NewCourtGroup.NonParallelReporters.Add 3
    NewCourtGroup.NonParallelReporters.Add 4

End Sub

Private Sub AddGa()
    Dim NewJur As cJurisdiction
    Dim NewCourtGroup As cCourtGroup
    Dim NewCourt As cCourt

    Set NewJur = AddJurisdiction("Georgia", "Ga.")

```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Georgia Supreme Court", "", 1846, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Georgia Court of Appeals", "Ct. App.", 1907, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddHaw()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Hawaii", "Haw.", "hi")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Hawaii Supreme Court", "", 1847, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Intermediate Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Hawaii Intermediate Court of Appeals", "Ct. App.", 1980, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddIdaho()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Idaho", "Idaho", "id")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Idaho Supreme Court", "", 1866, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Idaho Court of Appeals", "Ct. App.", 1982, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddIll()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Illinois", "Ill.", "il")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Illinois Supreme Court", "", 1819, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
NewCourtGroup.ParallelReporters.Add 3

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Appellate Court")
Set NewCourt = AddCourt(NewCourtGroup, "Illinois Appellate Court", "App. Ct.", 1877, 0, False, "ctapp", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
NewCourtGroup.ParallelReporters.Add 3

```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Claims")
Set NewCourt = AddCourt(NewCourtGroup, "Illinois Court of Claims", "Ct. Cl.", 1889, 0, False)
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddInd()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Indiana", "Ind.", "in")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Indiana Supreme Court", "", 1817, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Indiana Court of Appeals", "Ct. App.", 1890, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddIowa()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Iowa", "Iowa", "ia")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Iowa Supreme Court", "", 1855, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Iowa Court of Appeals", "Ct. App.", 1977, 0, False, "appct", "app")
NewCourtGroup.NonParallelReporters.Add 2
```

```
End Sub
```

```
Private Sub AddKan()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Kansas", "Kan.", "ks")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Kansas Supreme Court", "", 1858, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Kansas Court of Appeals", "Ct. App.", 1895, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddKy()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Kentucky", "Ky.")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Kentucky Supreme Court", "", 1976, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "Kentucky Court of Appeals", "", 1785, 1975, False, "appct", "app")
```

```

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Kentucky Court of Appeals", "Ct. App.", 1976, 0, False, "appct", "app")

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 3

```

End Sub

Private Sub AddLa()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Louisiana", "La.")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Louisiana Supreme Court", "", 1813, 0, False, "suprct", "supct", "sct")
Set NewCourt = AddCourt(NewCourtGroup, "Superior Court of Louisiana", "", 1809, 1812, False)
Set NewCourt = AddCourt(NewCourtGroup, "Superior Court of the Territory of Orleans", "Orleans", 1809, 1812, True)

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Louisiana Court of Appeal", "Ct. App.", 1881, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddMe()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Maine", "Me.")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Judicial Court")
Set NewCourt = AddCourt(NewCourtGroup, "Maine Supreme Judicial Court", "", 1820, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddMd()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Maryland", "Md.")

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Maryland Court of Appeals", "", 1770, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Special Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Maryland Court of Special Appeals", "Ct. Spec. App.", 1967, 0, False)

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddMass()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Massachusetts", "Mass.", "ma")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Judicial Court")

```

```

Set NewCourt = AddCourt(NewCourtGroup, "Massachusetts Supreme Judicial Court", "", 1805, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Appeals Court")
Set NewCourt = AddCourt(NewCourtGroup, "Massachusetts Appeals Court", "App. Ct.", 1972, 0, False, "ctapp", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "District Court")
Set NewCourt = AddCourt(NewCourtGroup, "Massachusetts District Court", "Dist. Ct.", 1936, 0, False)

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 3

```

End Sub

Private Sub AddMich()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Michigan", "Mich.", "mi")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Michigan Supreme Court", "", 1805, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Michigan Court of Appeals", "Ct. App.", 1965, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Claims")
Set NewCourt = AddCourt(NewCourtGroup, "Michigan Court of Claims", "Ct. Cl.", 1938, 1942, False)
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddMinn()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Minnesota", "Minn.", "mn")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Minnesota Supreme Court", "", 1851, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Minnesota Court of Appeals", "Ct. App.", 1983, 0, False, "appct", "app")
NewCourtGroup.NonParallelReporters.Add 2

```

End Sub

Private Sub AddMiss()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Mississippi", "Miss.", "ms")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Mississippi Supreme Court", "", 1818, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3

```

End Sub

Private Sub AddMo()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Missouri", "Mo.", "mo")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Missouri Supreme Court", "", 1821, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Missouri Court of Appeals", "Ct. App.", 1876, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddMont()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Montana", "Mont.", "mt")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Montana Supreme Court", "", 1868, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3

```

End Sub

Private Sub AddNeb()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Nebraska", "Neb.", "ne")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Nebraska Supreme Court", "", 1860, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Nebraska Court of Appeals", "Ct. App.", 1992, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddNev()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Nevada", "Nev.", "nv")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Nevada Supreme Court", "", 1865, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddNH()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("New Hampshire", "N.H.", "nh")

```



```

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "New Hampshire Supreme Court", "", 1816, 0, False, "suprct", "supct", "sct")

```

```

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

```

```

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddNJ()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

```

```
Set NewJur = AddJurisdiction("New Jersey", "N.J.")
```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey Supreme Court", "", 1790, 0, False)

```

```

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

```

```

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Superior Court")
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey Superior Court", "Super. Ct. App. Div.", 1790, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey Superior Court", "Super. Ct. Ch. Div.", 1790, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey Superior Court", "Super. Ct. Law Div.", 1790, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey Court of Chancery", "Ch.", 1790, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey Supreme Court", "Sup. Ct.", 1790, 0, False, "suprct", "supct", "sct")
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey Perogative Court", "Perog. Ct.", 1790, 0, False)

```

```

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

```

```

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "County Court")
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey County Court", "County Ct.", 1790, 0, False)

```

```

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

```

```

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Tax Court")
Set NewCourt = AddCourt(NewCourtGroup, "New Jersey Tax Court", "Tax Ct.", 1979, 0, False)
Set NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddNM()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

```

```
Set NewJur = AddJurisdiction("New Mexico", "N.M.")
```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "New Mexico Supreme Court", "", 1883, 0, False, "suprct", "supct", "sct")

```

```

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

```

```

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "New Mexico Court of Appeals", "Ct. App.", 1967, 0, False, "appct", "app")

```

```

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

```

```

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

```

End Sub

Private Sub AddNY()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

```

```
Set NewJur = AddJurisdiction("New York", "N.Y.")
```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "New York Court of Appeals", "", 1847, 0, False, "ctapp", "app", "appct", "app")

```

```

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
NewCourtGroup.ParallelReporters.Add 3

```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court for the Correction of Errors")
Set NewCourt = AddCourt(NewCourtGroup, "New York Court for the Correction of Errors", "", 1791, 1847, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York Supreme Court of Judicature", "Sup. Ct.", 1791, 1847, False, "suprct", "supct", "sct")
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Chancery")
Set NewCourt = AddCourt(NewCourtGroup, "New York Court of Chancery", "Ch.", 1814, 1847, False)
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court, Appellate Division")
Set NewCourt = AddCourt(NewCourtGroup, "New York Supreme Court, Appellate Division", "App. Div.", 1847, 0, False, "sctappdiv", "supctapdiv")
```

```
NewCourtGroup.ParallelReporters.Add 3
NewCourtGroup.ParallelReporters.Add 4
```

```
NewCourtGroup.NonParallelReporters.Add 4
NewCourtGroup.NonParallelReporters.Add 3
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Other lower courts")
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts", "App. Term.", 1888, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts", "Sup. Ct.", 1888, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts", "Ct. Cl.", 1888, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts", "Civ. Ct.", 1888, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts", "Crim. Ct.", 1888, 0, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts", "Fam. Ct.", 1888, 0, False)
```

```
NewCourtGroup.ParallelReporters.Add 3
NewCourtGroup.ParallelReporters.Add 4
```

```
NewCourtGroup.NonParallelReporters.Add 4
NewCourtGroup.NonParallelReporters.Add 3
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Other lower courts before 1888")
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts before 1888", "App. Term.", 1844, 1887, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts before 1888", "Sup. Ct.", 1844, 1887, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts before 1888", "Ct. Cl.", 1844, 1887, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts before 1888", "Civ. Ct.", 1844, 1887, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts before 1888", "Crim. Ct.", 1844, 1887, False)
Set NewCourt = AddCourt(NewCourtGroup, "New York other lower courts before 1888", "Fam. Ct.", 1844, 1887, False)
NewCourtGroup.NonParallelReporters.Add 3
```

```
End Sub
```

```
Private Sub AddNC()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("North Carolina", "N.C.", "ncarolina")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "North Carolina Supreme Court", "", 1778, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "North Carolina Court of Appeals", "Ct. App.", 1968, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddND()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("North Dakota", "N.D.", "ndakota")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "North Dakota Supreme Court", "", 1890, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court of Dakota")
Set NewCourt = AddCourt(NewCourtGroup, "Supreme Court of Dakota", "Dakota", 1867, 1889, True)
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals of North Dakota")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals of North Dakota", "Ct. App.", 1987, 0, False, "appct", "app")
NewCourtGroup.NonParallelReporters.Add 2

```

End Sub

Private Sub AddOhio()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Ohio", "Ohio", "oh")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Ohio Supreme Court", "", 1821, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Ohio Court of Appeals", "Ct. App.", 1913, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Other law courts")
Set NewCourt = AddCourt(NewCourtGroup, "Other law courts", "", 1840, 0, False)

NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3
NewCourtGroup.NonParallelReporters.Add 4
NewCourtGroup.NonParallelReporters.Add 5
NewCourtGroup.NonParallelReporters.Add 6
NewCourtGroup.NonParallelReporters.Add 7
NewCourtGroup.NonParallelReporters.Add 8
NewCourtGroup.NonParallelReporters.Add 9
NewCourtGroup.NonParallelReporters.Add 10
NewCourtGroup.NonParallelReporters.Add 11
NewCourtGroup.NonParallelReporters.Add 12
NewCourtGroup.NonParallelReporters.Add 13

```

End Sub

Private Sub AddOkla()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Oklahoma", "Okla.", "ok")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Oklahoma Supreme Court", "", 1890, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals of Indian Territory")
Set NewCourt = AddCourt(NewCourtGroup, "Court of Appeals of Indian Territory", "Indian Terr.", 1896, 1907, True, "indterr")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Criminal Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Oklahoma Court of Criminal Appeals", "Crim. App.", 1959, 0, False, "ctcrimapp")
Set NewCourt = AddCourt(NewCourtGroup, "Oklahoma Criminal Court of Appeals", "Crim. App.", 1908, 1958, False, "ctcrimapp")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Oklahoma Court of Appeals", "Ct. App.", 1971, 0, False, "appct", "app")

NewCourtGroup.NonParallelReporters.Add 2

```

End Sub

Private Sub AddOr()

```

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Oregon", "Or.")

```

```

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Oregon Supreme Court", "", 1853, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Oregon Court of Appeals", "Ct. App.", 1969, 0, False, "appct", "app")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Tax Court")
Set NewCourt = AddCourt(NewCourtGroup, "Oregon Tax Court", "T.C.", 1962, 0, False, "taxct", "taxcourt")

NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddPa()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Pennsylvania", "Pa.", "penn")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Pennsylvania Supreme Court", "", 1754, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Superior Court")
Set NewCourt = AddCourt(NewCourtGroup, "Pennsylvania Superior Court", "Super. Ct.", 1895, 0, False)

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Commonwealth Court")
Set NewCourt = AddCourt(NewCourtGroup, "Pennsylvania Commonwealth Court", "Commw. Ct.", 1970, 0, False)

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

Set NewCourtGroup = AddCourtGroup(NewJur, "Other lower courts")
Set NewCourt = AddCourt(NewCourtGroup, "Pennsylvania other lower courts", "", 1870, 0, False)

NewCourtGroup.NonParallelReporters.Add 1
NewCourtGroup.NonParallelReporters.Add 3

End Sub

Private Sub AddRI()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("Rhode Island", "R.I.")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Rhode Island Supreme Court", "", 1828, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddSC()

Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt

Set NewJur = AddJurisdiction("South Carolina", "S.C.", "scarolina")

Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "South Carolina Supreme Court", "", 1868, 0, False, "suprct", "supct", "sct")

NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2

```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "South Carolina Court of Appeals", "Ct. App.", 1983, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Courts of law")
Set NewCourt = AddCourt(NewCourtGroup, "South Carolina Courts of law", "L.", 1783, 1868, False, "law")
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Courts of equity")
Set NewCourt = AddCourt(NewCourtGroup, "South Carolina courts of equity", "Eq.", 1784, 1868, False, "equity")
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddSD()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("South Dakota", "S.D.", "sdakota")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "South Dakota Supreme Court", "", 1890, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court of Dakota")
Set NewCourt = AddCourt(NewCourtGroup, "Supreme Court of Dakota", "Dakota", 1867, 1889, True)
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddTenn()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Tennessee", "Tenn.", "tn")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Tennessee Supreme Court", "", 1791, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Tennessee Court of Appeals", "Ct. App.", 1925, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Criminal Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Tennessee Court of Criminal Appeals", "Crim. App.", 1967, 0, False, "ctcrimapp")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddTex()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Texas", "Tex.", "tx")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Texas Supreme Court", "", 1840, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Criminal Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Texas Court of Criminal Appeals", "Crim. App.", 1891, 0, False, "ctcrimapp")
Set NewCourt = AddCourt(NewCourtGroup, "Texas Court of Appeals", "Ct. App.", 1876, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Commission of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Texas Commission of Appeals", "Comm'n App.", 1879, 0, False)
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Courts of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Texas Courts of Appeals", "App.", 1892, 0, False)
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

End Sub

Private Sub AddUtah()

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Utah", "Utah", "ut")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Utah Supreme Court", "", 1851, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Utah Court of Appeals", "Ct. App.", 1987, 0, False, "appct", "app")
```

```
NewCourtGroup.NonParallelReporters.Add 2
```

End Sub

Private Sub AddVt()

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Vermont", "Vt.")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Vermont Supreme Court", "", 1789, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

End Sub

Private Sub AddVa()

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Virginia", "Va.")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
Set NewCourt = AddCourt(NewCourtGroup, "Virginia Supreme Court", "", 1779, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
Set NewCourt = AddCourt(NewCourtGroup, "Virginia Court of Appeals", "Ct. App.", 1985, 0, False, "appct", "app")
```

```
NewCourtGroup.ParallelReporters.Add 1
NewCourtGroup.ParallelReporters.Add 2
```

```
NewCourtGroup.NonParallelReporters.Add 2
```

```

NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddWash()

    Dim NewJur As cJurisdiction
    Dim NewCourtGroup As cCourtGroup
    Dim NewCourt As cCourt

    Set NewJur = AddJurisdiction("Washington", "Wash.", "wn", "wa")

    Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
    Set NewCourt = AddCourt(NewCourtGroup, "Washington Supreme Court", "", 1854, 0, False, "suprct", "supct", "sct")

    NewCourtGroup.ParallelReporters.Add 1
    NewCourtGroup.ParallelReporters.Add 2

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 1

    Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
    Set NewCourt = AddCourt(NewCourtGroup, "Washington Court of Appeals", "Ct. App.", 1969, 0, False, "appct", "app")

    NewCourtGroup.ParallelReporters.Add 1
    NewCourtGroup.ParallelReporters.Add 2

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddWVa()

    Dim NewJur As cJurisdiction
    Dim NewCourtGroup As cCourtGroup
    Dim NewCourt As cCourt

    Set NewJur = AddJurisdiction("West Virginia", "W. Va.", "wv", "wvirginia")

    Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court of Appeals")
    Set NewCourt = AddCourt(NewCourtGroup, "West Virginia Supreme Court of Appeals", "", 1864, 0, False, "suprct", "supct", "sct", "sctapp")

    NewCourtGroup.ParallelReporters.Add 1
    NewCourtGroup.ParallelReporters.Add 2

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddWis()

    Dim NewJur As cJurisdiction
    Dim NewCourtGroup As cCourtGroup
    Dim NewCourt As cCourt

    Set NewJur = AddJurisdiction("Wisconsin", "Wis.", "wi")

    Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
    Set NewCourt = AddCourt(NewCourtGroup, "Wisconsin Supreme Court", "", 1839, 0, False, "suprct", "supct", "sct")

    NewCourtGroup.ParallelReporters.Add 1
    NewCourtGroup.ParallelReporters.Add 2

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 1

    Set NewCourtGroup = AddCourtGroup(NewJur, "Court of Appeals")
    Set NewCourt = AddCourt(NewCourtGroup, "Wisconsin Court of Appeals", "Ct. App.", 1978, 0, False, "appct", "app")

    NewCourtGroup.ParallelReporters.Add 1
    NewCourtGroup.ParallelReporters.Add 2

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddWyo()

    Dim NewJur As cJurisdiction
    Dim NewCourtGroup As cCourtGroup
    Dim NewCourt As cCourt

    Set NewJur = AddJurisdiction("Wyoming", "Wyo.", "wy")

    Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
    Set NewCourt = AddCourt(NewCourtGroup, "Wyoming Supreme Court", "", 1870, 0, False, "suprct", "supct", "sct")

    NewCourtGroup.ParallelReporters.Add 1
    NewCourtGroup.ParallelReporters.Add 2

    NewCourtGroup.NonParallelReporters.Add 2
    NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Sub AddAmSamoa()

```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("American Samoa", "Am. Samoa")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "High Court of American Samoa")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "High Court of American Samoa", "", 1900, 0, False, "highct", "highcourt")
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddCZ()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Canal Zone", "C.Z.")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "United States District Court for the Eastern District of Louisiana")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court for the Eastern District of Louisiana", "E.D. La.", 1982, 0, True)
NewCourtGroup.NonParallelReporters.Add 3
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "United States District Court for the District of the Canal Zone")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "U.S. District Court for the District of the Canal Zone", "D.C.Z.", 1846, 1982, True)
NewCourtGroup.NonParallelReporters.Add 3
```

```
End Sub
```

```
Private Sub AddGuam()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Guam", "Guam")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "District Court of Guam, Appellate Division")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "District Court of Guam, Appellate Division", "D. Guam App. Div.", 1951, 0, True)
```

```
NewCourtGroup.NonParallelReporters.Add 3
```

```
NewCourtGroup.NonParallelReporters.Add 4
```

```
End Sub
```

```
Private Sub AddNavajo()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Navajo Nation", "Navajo", "navajonat")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "Navajo Nation Supreme Court", "", 1969, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "District Court")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "Navajo Nation District Court", "D. Ct.", 1969, 0, False, "distct")
```

```
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddNMarI()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Northern Mariana Islands", "N. Mar. I.", "nmi")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "Northern Mariana Islands Supreme Court", "", 1989, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.NonParallelReporters.Add 1
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "District Court")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "District Court for the Northern Mariana Islands, Trial Division", "D. N. Mar. I.", 1979, 0, True)
```

```
Set NewCourt = AddCourt(NewCourtGroup, "District Court for the Northern Mariana Islands, Appellate Division", "D. N. Mar. I. App. Div.", 1979, 0, True)
```

```
Set NewCourt = AddCourt(NewCourtGroup, "Northern Mariana Islands Commonwealth Superior Court", "Commw. Super. Ct.", 1979, 0, False, "superct")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "Northern Mariana Islands Commonwealth Trial Court", "Commw. Trial Ct.", 1979, 0, False, "trialct")
```

```
NewCourtGroup.NonParallelReporters.Add 1
```

```
End Sub
```

```
Private Sub AddPR()
```

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

```
Set NewJur = AddJurisdiction("Puerto Rico", "P.R.")
```

```
Set NewCourtGroup = AddCourtGroup(NewJur, "Supreme Court")
```

```
Set NewCourt = AddCourt(NewCourtGroup, "Puerto Rico Supreme Court", "", 1899, 0, False, "suprct", "supct", "sct")
```

```
NewCourtGroup.ParallelReporters.Add 1
```


NewCourtGroup.ParallelReporters

End Sub

Private Sub AddVI()

```
Dim NewJur As cJurisdiction
Dim NewCourtGroup As cCourtGroup
Dim NewCourt As cCourt
```

Set NewJur = AddJurisdiction("Virgin Islands", "V.I.")

Set NewCourtGroup = AddCourtGroup(NewJur, "All courts")

Set NewCourt = AddCourt(NewCourtGroup, "Virgin Islands, all courts", "", 1917, 0, False)

NewCourtGroup.NonParallelReporters.Add 1

End Sub

Private Function AddJurisdiction(FullName As String, AbrvName As String, ParamArray Aliases()) As cJurisdiction

Dim varAliases As Variant

Set AddJurisdiction = New cJurisdiction

With AddJurisdiction

.FullName = FullName

.AbrvName = AbrvName

End With

varAliases = Aliases()

mbcJurisdictions.Add AddJurisdiction, varAliases

End Function

Private Function AddCourtGroup(ParentJurisdiction As cJurisdiction, ByVal FullName As String) As cCourtGroup

Set AddCourtGroup = New cCourtGroup

With AddCourtGroup

.FullName = FullName

Set .Parent = ParentJurisdiction

End With

ParentJurisdiction.CourtGroups.Add AddCourtGroup, FullName

mbcJurisdictions.AllCourtGroups.Add AddCourtGroup

End Function

Private Function AddCourt(ByRef CourtGroup As cCourtGroup, ByRef FullName As String, ByRef AbrvName As String, _
ByRef YearStart As Long, ByRef YearEnd As Long, ByRef JurisdictionImplicit As Boolean, ParamArray Aliases()) As cCourt

Dim varAliases As Variant

Set AddCourt = New cCourt

With AddCourt

.FullName = FullName

.AbrvName = AbrvName

.YearStart = YearStart

.JurisdictionImplicit = JurisdictionImplicit

If YearEnd > 0 Then .YearEnd = YearEnd Else .YearEnd = 9999

Set .CourtGroup = CourtGroup

Set .Parent = CourtGroup.Parent

End With

varAliases = Aliases()

AddCourt.Parent.Courts.Add AddCourt, varAliases

mbcJurisdictions.AllCourts.Add AddCourt, varAliases

End Function

cAutoSubs - 1

Option Explicit

Public Sub BCAutoExec()

```
Call InsertMenuItem("Tools", "BlueCheck Options...", "BlueCheckOptions", True)
Call InsertMenuItem("Tools", "&BlueCheck", "BlueCheck", False)
```

End Sub

Private Sub InsertMenuItem(MenuName As String, MenuItem As String, OnActionName As String, DividerAfter As Boolean)

```
Dim cbmMenuBar As CommandBarPopup
Dim cbbNewItem As CommandBarButton
Dim cbbFirstItem As CommandBarButton
Dim cbcExistingItem As CommandBarControl
```

' If the menu item already exists, then exit the sub.

```
Set cbcExistingItem = Word.CommandBars.FindControl(Tag:=MenuItem)
If Not cbcExistingItem Is Nothing Then Exit Sub
```

' If a divider bar should be added, then set the first item in the menu to a new group so that a divider will appear before it.

```
Set cbmMenuBar = Word.CommandBars("Menu Bar").Controls(MenuName)
```

```
If DividerAfter = True Then
    Set cbbFirstItem = cbmMenuBar.Controls(1)
    cbbFirstItem.BeginGroup = True
End If
```

' Add the menu item.

```
Set cbbNewItem = cbmMenuBar.Controls.Add(Type:=msoControlButton, Before:=1, temporary:=True)
```

```
With cbbNewItem
    .Caption = MenuItem
    .Tag = MenuItem
    .Enabled = True
    .OnAction = OnActionName
End With
```

End Sub

Public Sub BCAutoClose()

```
Call CloseAllHiddenDocuments
Call RemoveMenuItem("BlueCheck Options...")
Call RemoveMenuItem("&BlueCheck")
```

End Sub

Public Sub CloseAllHiddenDocuments()

```
Dim mswDocument As Word.Document
For Each mswDocument In Word.Documents
    If mswDocument.ActiveWindow.Visible = False Then
        mswDocument.Close wdDoNotSaveChanges
    End If
Next mswDocument
```

End Sub

Private Sub RemoveMenuItem(ItemToRemove As String)

```
Dim cbcExistingItem As CommandBarControl

Set cbcExistingItem = CommandBars.FindControl(Tag:=ItemToRemove)
If Not cbcExistingItem Is Nothing Then
    cbcExistingItem.Delete
End If
```

End Sub

161

BlueCheckMod - 1

'-- NOTE: "BlueCheck" and "BCWordIn" must be selected in the References dialog

162

Private mBlueCheck As New cBlueCheck

Public Sub AutoExec()

Call BCAutoExec

End Sub

Public Sub BlueCheck()

Call mBlueCheck.Start

Beep

End Sub

Public Sub BlueCheckOptions()

Call mBlueCheck.ShowOptions

End Sub

BlueCheckMod - 1
'-- NOTE: "BlueCheck" and "BCWordIn" must be selected in the References dialog
Private mBlueCheck As New cBlueCheck
Public Sub AutoExec()
Call BCAutoExec
End Sub
Public Sub BlueCheck()
Call mBlueCheck.Start
Beep
End Sub
Public Sub BlueCheckOptions()
Call mBlueCheck.ShowOptions
End Sub